4/17/15

Enterprise Application Development

LAB 06

Objectives:

The objective of this lab is to make you familiar with

- WPF data binding ,data grid and reports

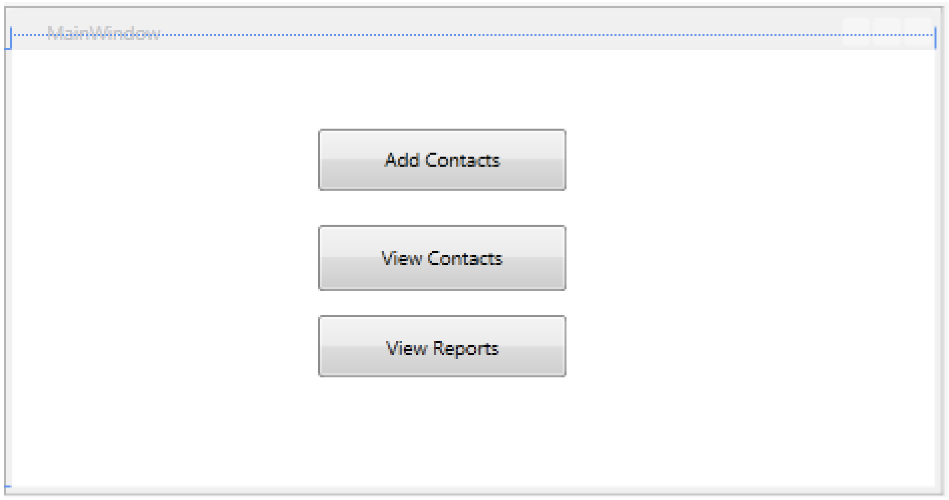
Task 1:

You are required to make a contact diary in which user can

- Save multiple contacts simultaneously
- View all contacts
- View Report

Solution 1:

- Make WPF Project
- Add 3 WPF (windows) for add, view and report forms.
- On home screen make a menu for add contact, view list and view report functionality.



When user will click on any button the respective form will open.

- Add a .mdf file for database and create a table Contact (id, fname, lname, email, job, stAddress, city, province)

Add contact:

Here use DataGrid to take input of multiple contacts simultaneously.

In Xaml (.xaml file):

Add the following code to generate datagrid. Here we will perform data binding using xaml tags.

```
 <Grid>
     <Label Content="Add Data" Margin="0,0,0,336" FontSize="20" FontWeight="Bold" />
      <DataGrid AlternatingRowBackground="LightBlue" x:Name="contactsGrid"
CanUserAddRows="True" IsReadOnly="False" AutoGenerateColumns="False"
CanUserDeleteRows="True" Margin="10,48,10,0" VerticalAlignment="Top" Height="265"  >
          <DataGrid.Columns>
             <DataGridTextColumn x:Name="c1"  Binding="{Binding fname}" Header="First Name"
IsReadOnly="False" Width="170"/>
             <DataGridTextColumn x:Name="c2"  Binding="{Binding lname}" Header="Last Name"
IsReadOnly="False" Width="170"/>
             <DataGridTextColumn x:Name="c3" Header="Email" Binding="{Binding email}"
IsReadOnly="False" Width="170"/>
             <DataGridTextColumn x:Name="c4" Header="Job" Binding="{Binding job}" IsReadOnly="False"
Width="170"/>
             <DataGridTextColumn x:Name="c5" Header="Street Address" Binding="{Binding stAddress}"
IsReadOnly="False" Width="170"/>
             <DataGridTextColumn x:Name="c6" Header="City" Binding="{Binding city}" IsReadOnly="False"
Width="170"/>
             <DataGridTextColumn x:Name="c7" Header="Province" Binding="{Binding province}"
IsReadOnly="False" Width="170"/>

          </DataGrid.Columns>
      </DataGrid>
      <Button Content="Save" HorizontalAlignment="Left" Margin="1010,334,0,0"
VerticalAlignment="Top" Width="109" Click="Button_Click_1" />
   </Grid>
```

Here I have created a

- label tag to display the text "Add Data"
- DataGrid tag to generate datagrid with the following properties
    1. AlternativeRowBackground="LightBlue" // it will set the background of alternative rows to light blue color
    2. CanUserAddRows="True" //it will let the user to add rows /record in your data grid.
    3. IsReadOnyl="False" //it will let the user to edit the datagrid so that he can add the new records.
    4. CanUserDeleteRows="True" // now user can delete rows while editing datagrid.
    5. AutoGenerateColumn="False" // it prevent the automatic generation of column in datagrid.

- Now I have added some column in the data grid along with the following properties.
  1. x:Name // this set the name of column
  2. Header // this is the text displayed on the head of the column
  3. Binding"{Binding fname}" //this is the most important property.It binds the column to a variable. reads the value in the corresponding variable.
  4. isReadOnly="False" //again it will let the user to edit the column
- In the last I have added a button to save the data.

In Code Behind (.cs)

- Make a class names ContactList with the attribute equals to the binding variable in xaml file.

```
class contactList
{
    public string fname { get; set; }
    public string lname { get; set; }
    public string job { get; set; }
    public string email { get; set; }
    public string stAddress { get; set; }
    public string city { get; set; }
    public string province { get; set; }
}
```

Here each attribute corresponds to a binding variable of a column.

- Now make a list of contactList and initialize it in constructor.

```
List<contactList> list;
list = new List<contactList>();
```

- Now make this list the itemSource of DataGrid also in constructor.

```
contactsGrid.ItemsSource = list;
```

- Override the click event of Button and write down the following code there.

```
private void Button_Click_1(object sender, RoutedEventArgs e)
    {
        for (int rows = 0; rows < contactsGrid.Items.Count - 1; rows++)
        {
            contactList conList= new contactList();
            conList = (contactList)contactsGrid.Items[rows];
            saveData(conList);
        }
    }
```

// this for loop will read all rows of data grid one by one in contactList objects because we make contactList a datasource of DataGrid

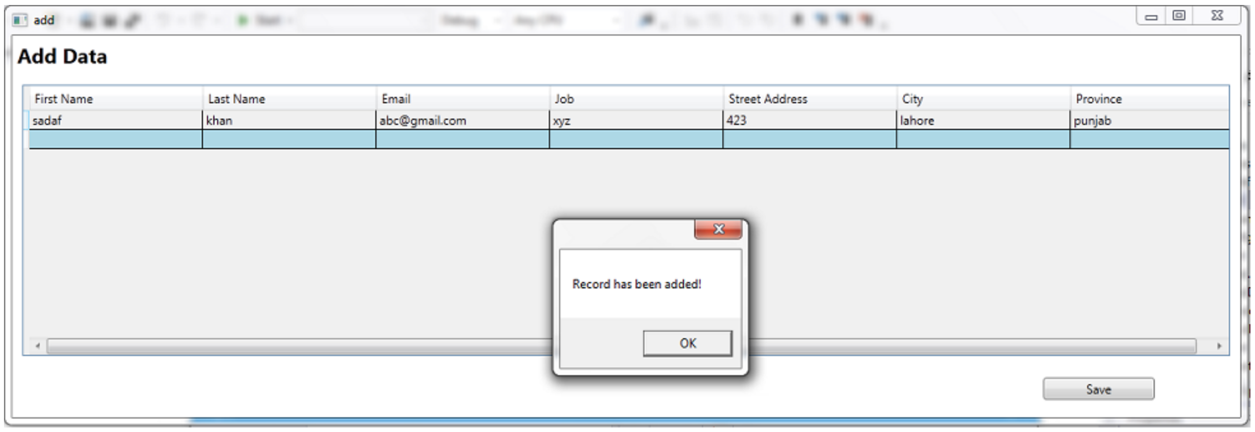
//now call saveData Function to save data in database.

```
private void saveData(contactList obj)
{
```

```
String connection = @"Data Source=(LocalDB)\v11.0;AttachDbFilename=C:\Users\Sadaf
Khan\Desktop\Lab 06\Lab 06\Database1.mdf;Integrated Security=True";

string q = "insert into ContactList (fname,lname,email,job,stAddress,city,province)
values(@f,@l,@e,@j,@s,@c,@p)";
SqlConnection c1 = new SqlConnection(connection);
SqlCommand cmd =  new SqlCommand(q,c1);
SqlParameter p1 = new SqlParameter("f",obj.fname);
SqlParameter p2 = new SqlParameter("l", obj.lname);
SqlParameter p3 = new SqlParameter("e", obj.email);
SqlParameter p4 = new SqlParameter("j", obj.job);
SqlParameter p5 = new SqlParameter("s", obj.stAddress);
SqlParameter p6 = new SqlParameter("c", obj.city);
SqlParameter p7 = new SqlParameter("p", obj.province);

cmd.Parameters.Add(p1);
cmd.Parameters.Add(p2);
cmd.Parameters.Add(p3);
cmd.Parameters.Add(p4);
cmd.Parameters.Add(p5);
cmd.Parameters.Add(p6);
cmd.Parameters.Add(p7);
c1.Open();
int i=cmd.ExecuteNonQuery();
if (i > 0)
{
MessageBox.Show("Record has been added!");
}
c1.Close();
}
```



Add Data

First Name	Last Name	Email	Job	Street Address	City	Province
sadaf	khan	abc@gmail.com	xyz	423	lahore	punjab

Record has been added!

4/17/15

View All Contacts:

In xaml (.xaml file):

- Just add a tag to create a datagrid. Now we will create column using code behind.

```
<Grid>
    <Label Content="View Data" FontSize="20" FontWeight="Bold"></Label>
    <DataGrid AlternatingRowBackground="LightBlue" x:Name="viewContactsGrid"
CanUserAddRows="False" IsReadOnly="True" AutoGenerateColumns="False"
CanUserDeleteRows="False" Margin="10,48,10,0" VerticalAlignment="Top"
Height="265"></DataGrid>
</Grid>
```

In Code Behind (.cs):

- Reuse contactList class for databinding.
- Add columns in datagrid also assign them the corresponding attributes of contactList class.

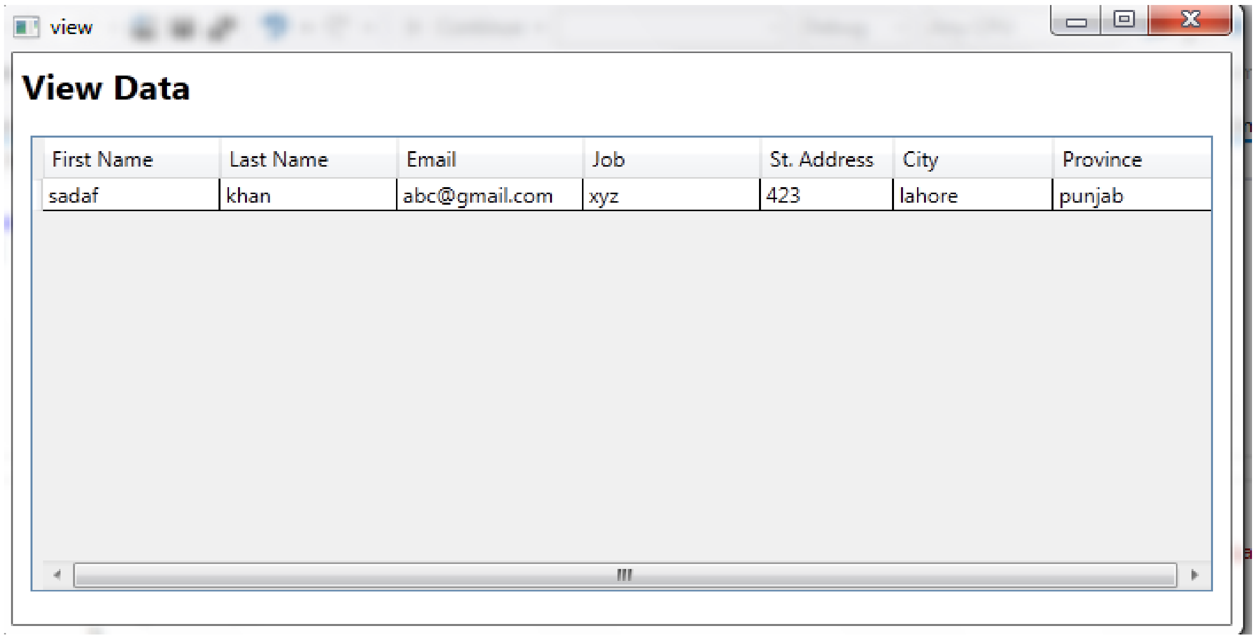
```
public void addColumnsInGrid()
  {
    if(viewContactsGrid.Columns.Count>0)
    {
      viewContactsGrid.Columns.Clear();
    }
    DataGridTextColumn c1 = new DataGridTextColumn();
    c1.Binding = new Binding("fname");
    c1.Header = "First Name";
    c1.Width = 100;
    DataGridTextColumn c3 = new DataGridTextColumn();
    c3.Binding = new Binding("lname");
    c3.Header = "Last Name";
    c3.Width = 100;
    DataGridTextColumn c2 = new DataGridTextColumn();
    c2.Binding = new Binding("email");
    c2.Header = "Email";
    c2.Width = 105;
    DataGridTextColumn c4 = new DataGridTextColumn();
    c4.Binding = new Binding("job");
    c4.Header = "Job";
    c4.Width = 100;
    DataGridTextColumn c5 = new DataGridTextColumn();
    c5.Binding = new Binding("stAddress");
    c5.Header = "St. Address";
    c5.Width = 75;
    DataGridTextColumn  c6 = new  DataGridTextColumn();
    c6.Binding = new Binding("city");
    c6.Header = "City";
    c6.Width = 90;
    DataGridTextColumn c7 = new DataGridTextColumn();
    c7.Binding = new Binding("province");
    c7.Header = "Province";
    c7.Width = 90;
    viewContactsGrid.Columns.Add(c1);
    viewContactsGrid.Columns.Add(c3);
    viewContactsGrid.Columns.Add(c2);
    viewContactsGrid.Columns.Add(c4);
    viewContactsGrid.Columns.Add(c5);
    viewContactsGrid.Columns.Add(c6);
    viewContactsGrid.Columns.Add(c7);
  }
```

1. DataGridTextColumn c=new DataGridTextColumn() // it creates a column
2. c.Binding // it sets the binding with the corresponding attribute in contactList class. The value of this attribute will be shown in this column.

- Now fetch data from database and show it in datagrid.

```
public void getData()
    {
                String connection = @"Data
                Source=(LocalDB)\v11.0;AttachDbFilename=C:\Users\Sadaf
                Khan\Desktop\Lab 06\Lab 06\Database1.mdf;Integrated
                Security=True";
        List<contactList> list = new List<contactList>();
        string q = "select * from ContactList";
        SqlConnection c1 = new SqlConnection(connection);
        SqlCommand cmd =  new SqlCommand(q,c1);
        c1.Open();
        SqlDataReader dr = cmd.ExecuteReader();
        while (dr.Read())
        {
          contactList obj = new contactList();
          obj.fname = dr[1].ToString();
          obj.lname = dr[2].ToString();
          obj.email = dr[3].ToString();
          obj.job = dr[4].ToString();
          obj.stAddress = dr[5].ToString();
          obj.city = dr[6].ToString();
          obj.province = dr[7].ToString();
          list.Add(obj);
        }
        viewContactsGrid.ItemsSource = list;
 c1.Close();
    }
```

- The line **viewContactsGrid.ItemsSource = list;** will show the data in grid.

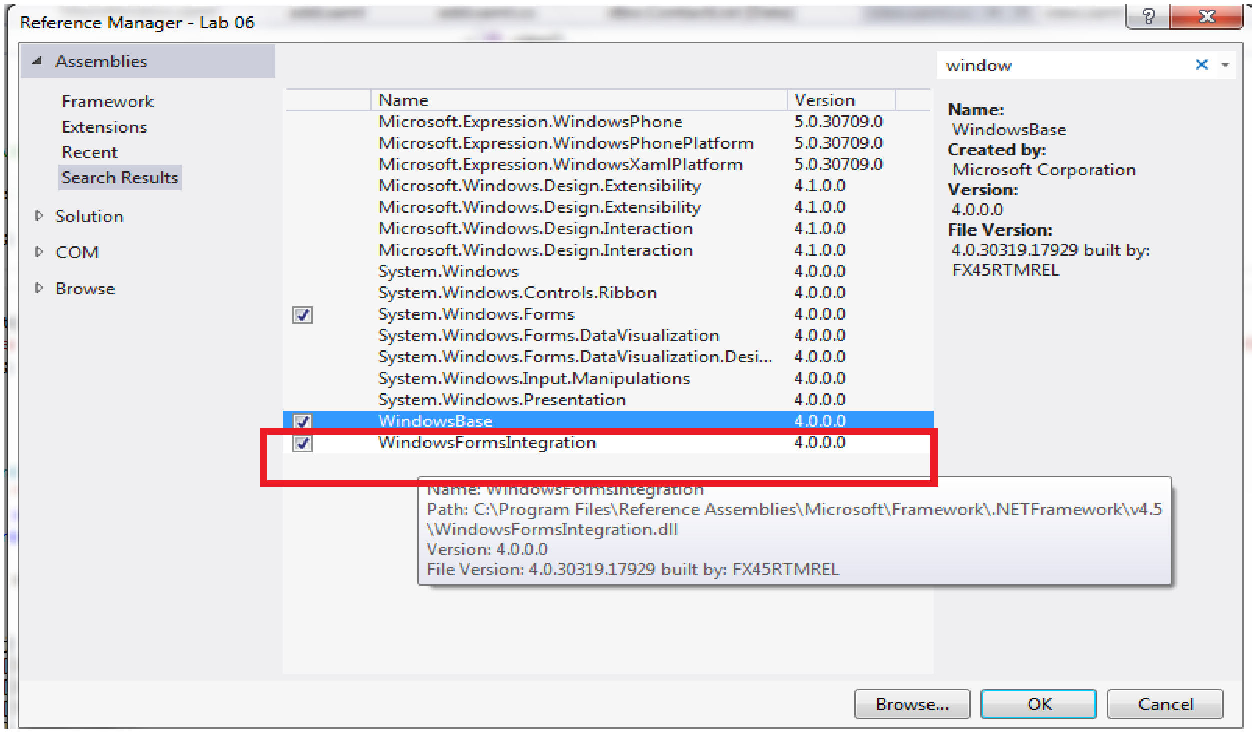


View Data

First Name	Last Name	Email	Job	St. Address	City	Province
sadaf	khan	abc@gmail.com	xyz	423	lahore	punjab

View Report:

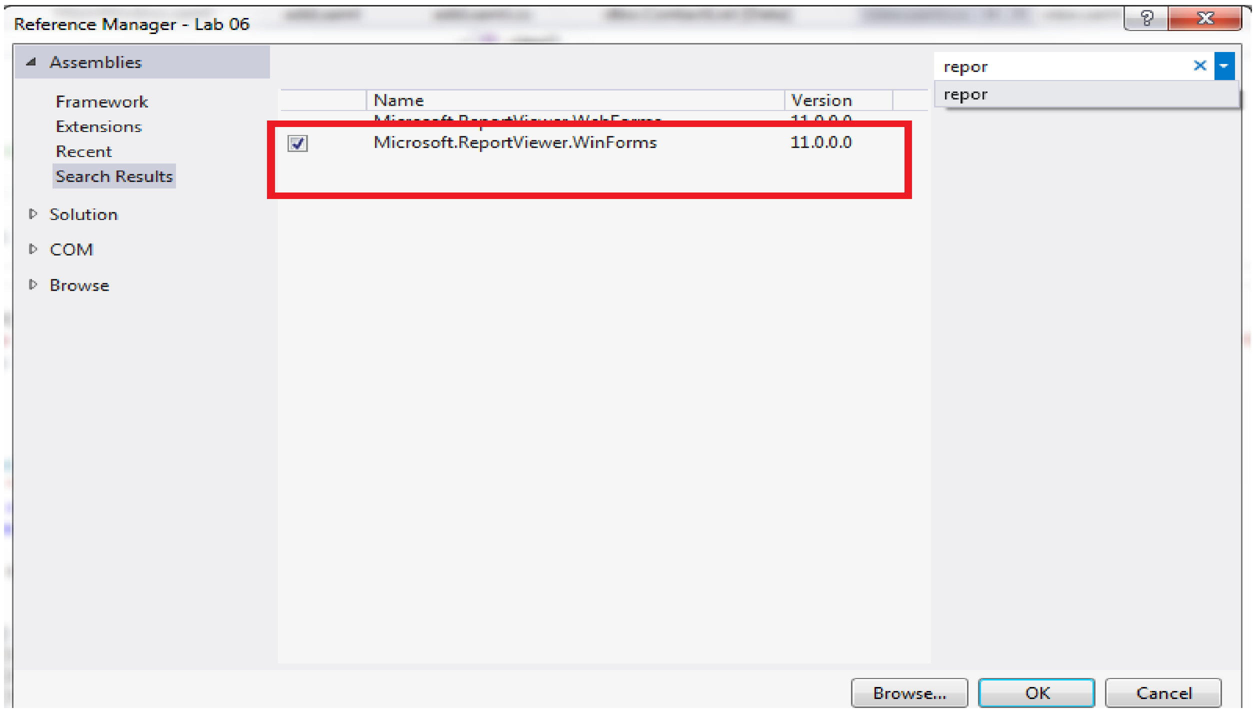
Here now first add 2 following references. Report viewer is not directly available in WPF. So you need to add references.



	Name	Version
	Microsoft.Expression.WindowsPhone	5.0.30709.0
	Microsoft.Expression.WindowsPhonePlatform	5.0.30709.0
	Microsoft.Expression.WindowsXamlPlatform	5.0.30709.0
	Microsoft.Windows.Design.Extensibility	4.1.0.0
	Microsoft.Windows.Design.Extensibility	4.1.0.0
	Microsoft.Windows.Design.Interaction	4.1.0.0
	Microsoft.Windows.Design.Interaction	4.1.0.0
	System.Windows	4.0.0.0
	System.Windows.Controls.Ribbon	4.0.0.0
✓	System.Windows.Forms	4.0.0.0
	System.Windows.Forms.DataVisualization	4.0.0.0
	System.Windows.Forms.DataVisualization.Desi...	4.0.0.0
	System.Windows.Input.Manipulations	4.0.0.0
	System.Windows.Presentation	4.0.0.0
✓	WindowsBase	4.0.0.0
✓	WindowsFormsIntegration	4.0.0.0

Name: WindowsBase
Created by: Microsoft Corporation
Version: 4.0.0.0
File Version: 4.0.30319.17929 built by: FX45RTMREL

Name: WindowsFormsIntegration
Path: C:\Program Files\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.5\WindowsFormsIntegration.dll
Version: 4.0.0.0
File Version: 4.0.30319.17929 built by: FX45RTMREL



Name	Version
☑ Microsoft.ReportViewer.WinForms	11.0.0.0

Now add the following line in your window tag

```
xmlns:rv="clr-namespace:Microsoft.Reporting.WinForms;assembly=Microsoft.ReportViewer.WinForms"
```

<u>In xaml (.xaml file):</u>

In Xaml file add the following code.

```
<Grid>
      <WindowsFormsHost  HorizontalAlignment="Left" Height="281" VerticalAlignment="Top"
Width="841" Margin="10,10,0,-21">
          <rv:ReportViewer x:Name="_reportViewer"  BackColor="White"/>
      </WindowsFormsHost>

   </Grid>
```

Now you know that Report viewer (used to view reports) is not available in WPF. So we host it by using WindowsFormsHost with the help of above added references.

<u>In Code Behind (.cs):</u>

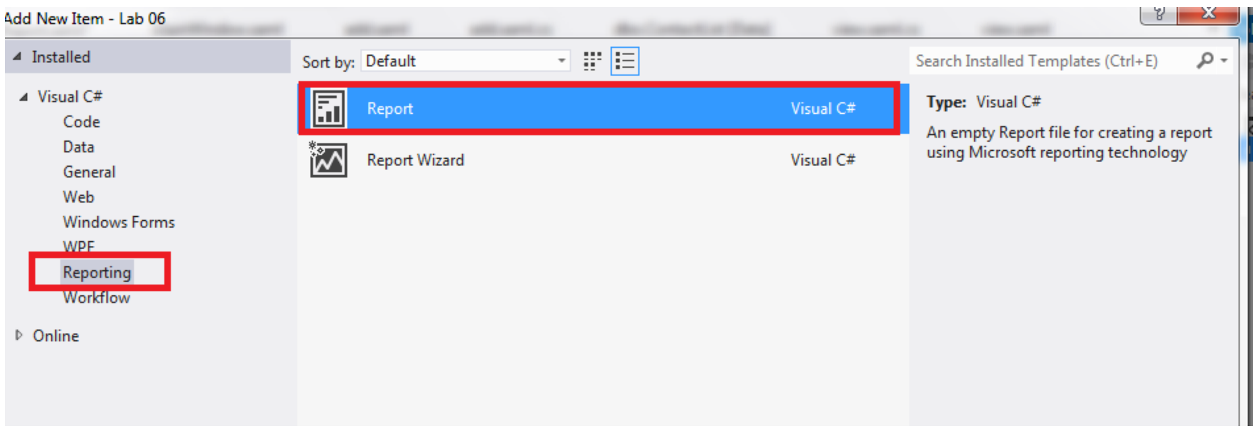
- Again resue the class contactList and read all data in List of Contact List.

```
public void getData()
{
          String connection = @"Data Source=(LocalDB)\v11.0;AttachDbFilename=C:\Users\Sadaf
          Khan\Desktop\Lab 06\Lab 06\Database1.mdf;Integrated Security=True";

        list = new List<contactList>();
        string q = "select * from ContactList";
```

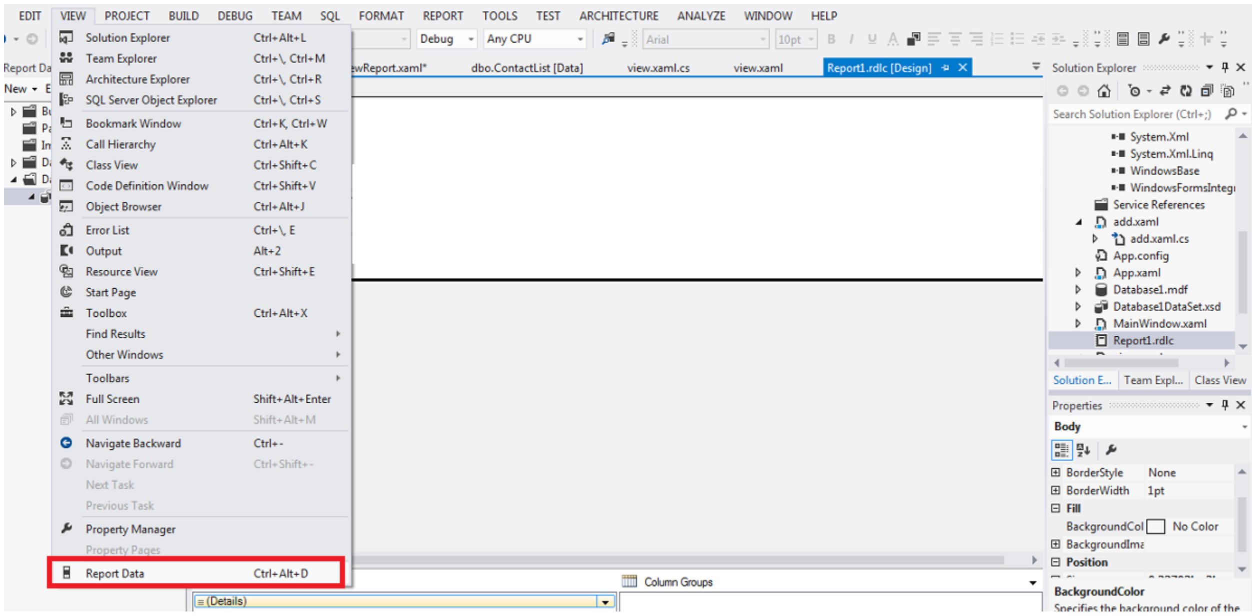
```
                SqlConnection c1 = new SqlConnection(connection);
                SqlCommand cmd = new SqlCommand(q, c1);
                c1.Open();
                SqlDataReader dr = cmd.ExecuteReader();
                while (dr.Read())
                {
                    contactList obj = new contactList();
                    obj.fname = dr[1].ToString();
                    obj.lname = dr[2].ToString();
                    obj.email = dr[3].ToString();
                    obj.job = dr[4].ToString();
                    obj.stAddress = dr[5].ToString();
                    obj.city = dr[6].ToString();
                    obj.province = dr[7].ToString();
                    list.Add(obj);
            }
                c1.Close();
          }
```

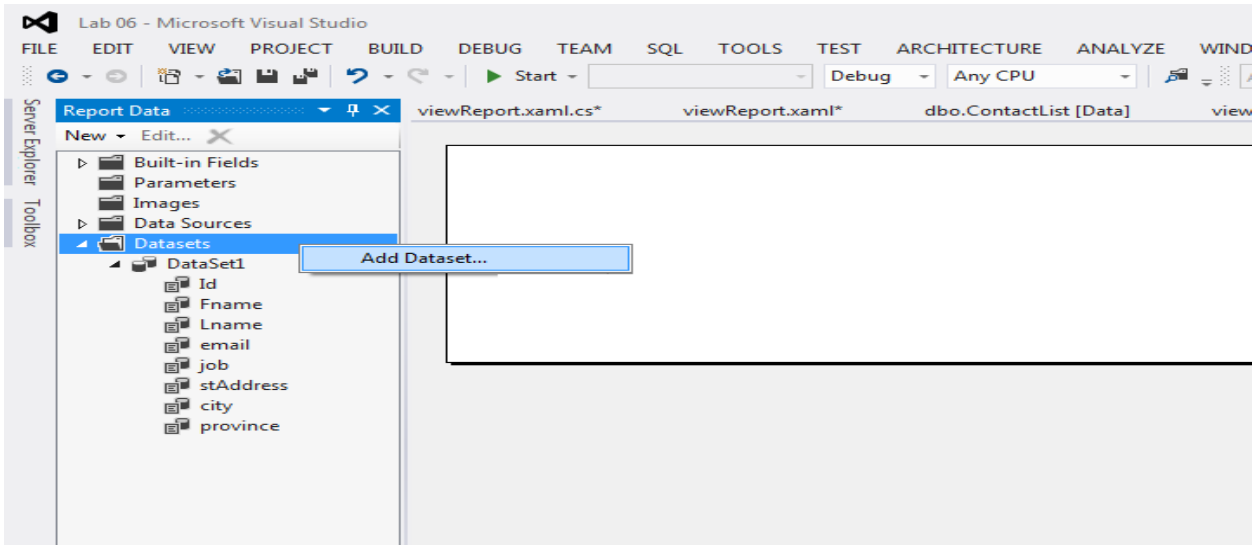
- After fetching all the data show it in your report viewer.
- For this purpose first add a ".rdlc report " into your project then sets it data source.
    1. Right click on project
    2. Select Reporting in Wizard
    3. Reporting->Report
    4. Click on ok button
    5. Now select report ,click on view tab and select report data
    6. Select Data source,right click on it ,select add new data source,
    7. Now click on new
    8. Select database
    9. Select datasource
    10. Next->next
    11. Check "tables"
    12. Click okay

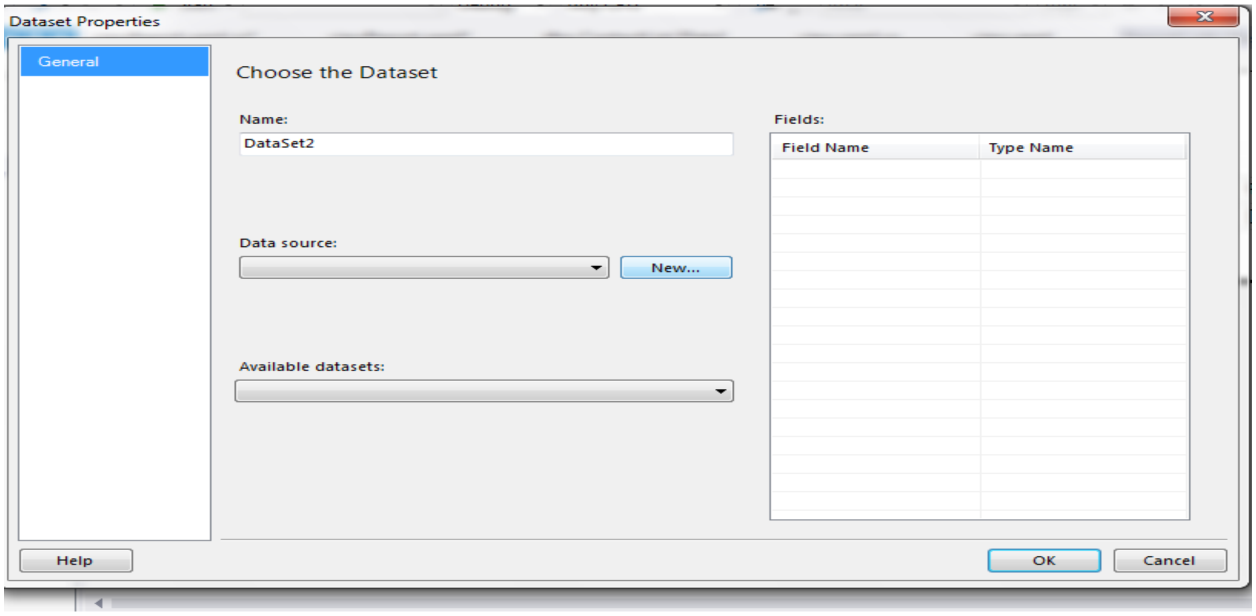


Type: Visual C#

An empty Report file for creating a report using Microsoft reporting technology


VIEW
Solution Explorer Ctrl+Alt+L
Team Explorer Ctrl+\, Ctrl+M
Architecture Explorer Ctrl+\, Ctrl+R
SQL Server Object Explorer Ctrl+\, Ctrl+S
Bookmark Window Ctrl+K, Ctrl+W
Call Hierarchy Ctrl+Alt+K
Class View Ctrl+Shift+C
Code Definition Window Ctrl+Shift+V
Object Browser Ctrl+Alt+J
Error List Ctrl+\, E
Output Alt+2
Resource View Ctrl+Shift+E
Start Page
Toolbox Ctrl+Alt+X
Find Results
Other Windows
Toolbars
Full Screen Shift+Alt+Enter
All Windows Shift+Alt+M
Navigate Backward Ctrl+-
Navigate Forward Ctrl+Shift+-
Next Task
Previous Task
Property Manager
Property Pages
Report Data Ctrl+Alt+D


Lab 06 - Microsoft Visual Studio
Report Data
New Edit...
Built-in Fields
Parameters
Images
Data Sources
Datasets
DataSet1
Id
Fname
Lname
email
job
stAddress
city
province
Add Dataset...


Dataset Properties

Choose the Dataset

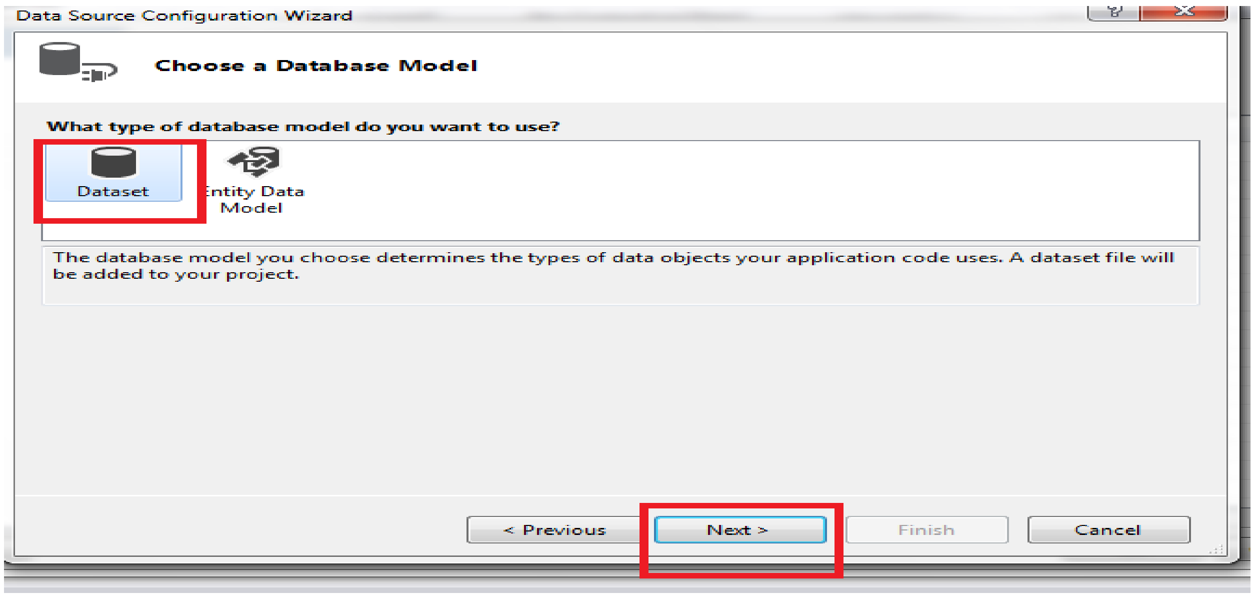
Name: DataSet2

Data source:

Available datasets:

Fields:

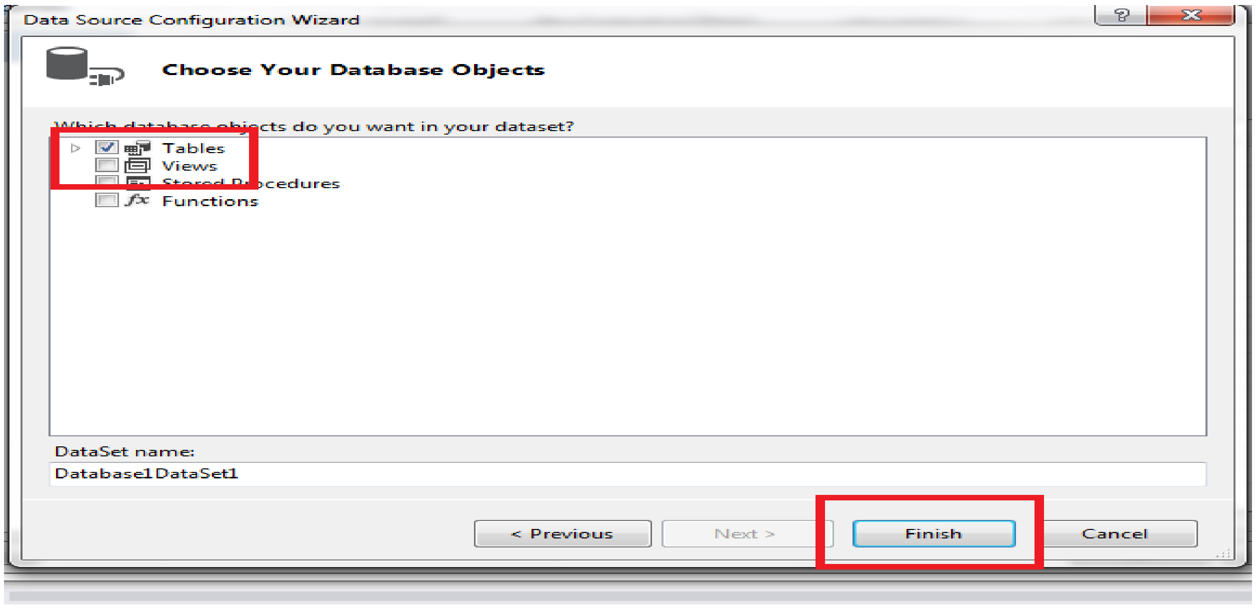
Field Name	Type Name


Data Source Configuration Wizard

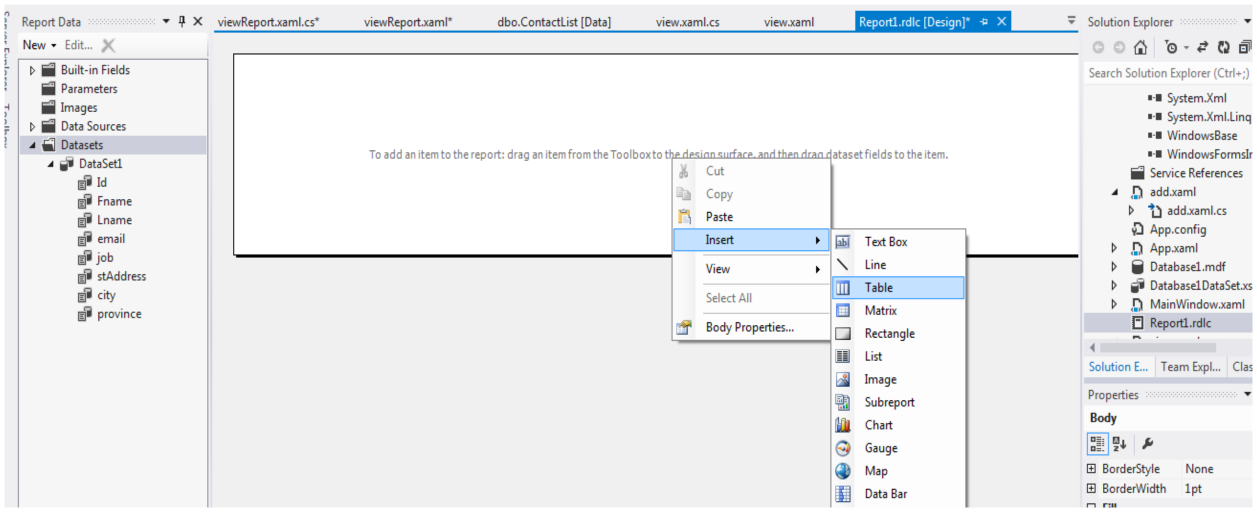
Choose a Database Model

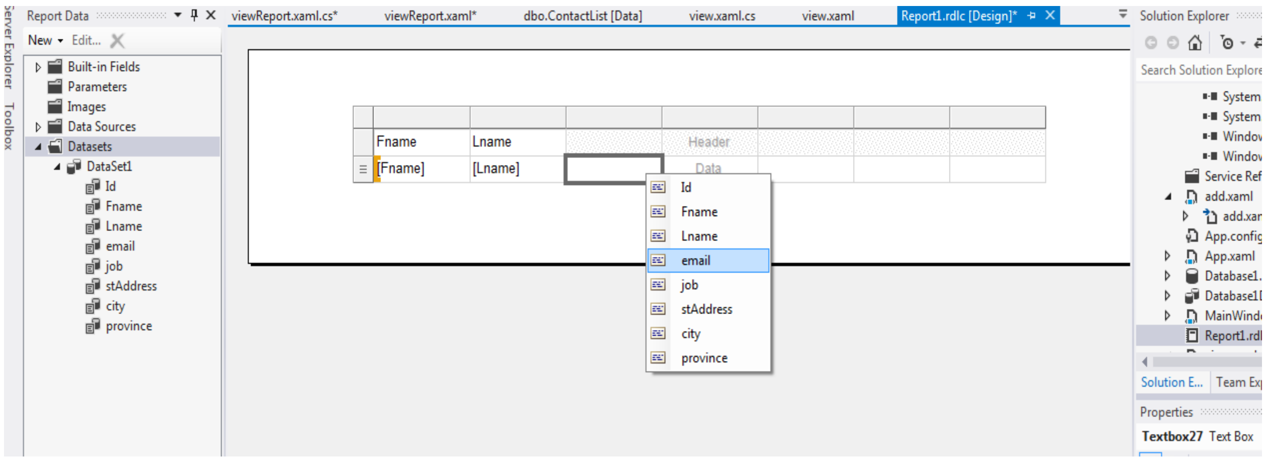
What type of database model do you want to use?

The database model you choose determines the types of data objects your application code uses. A dataset file will be added to your project.


Data Source Configuration Wizard
Choose Your Database Objects
Which database objects do you want in your dataset?
Tables
Views
Stored Procedures
Functions
DataSet name:
Database1DataSet1
< Previous
Next >
Finish
Cancel

- Now on report surface right click and add a table.
- After this select each column and add its datasource.
- Right click on the surface of report and add a text box.
- Add the contact “Contact List Report” in text box
- Rename the columns of reports.



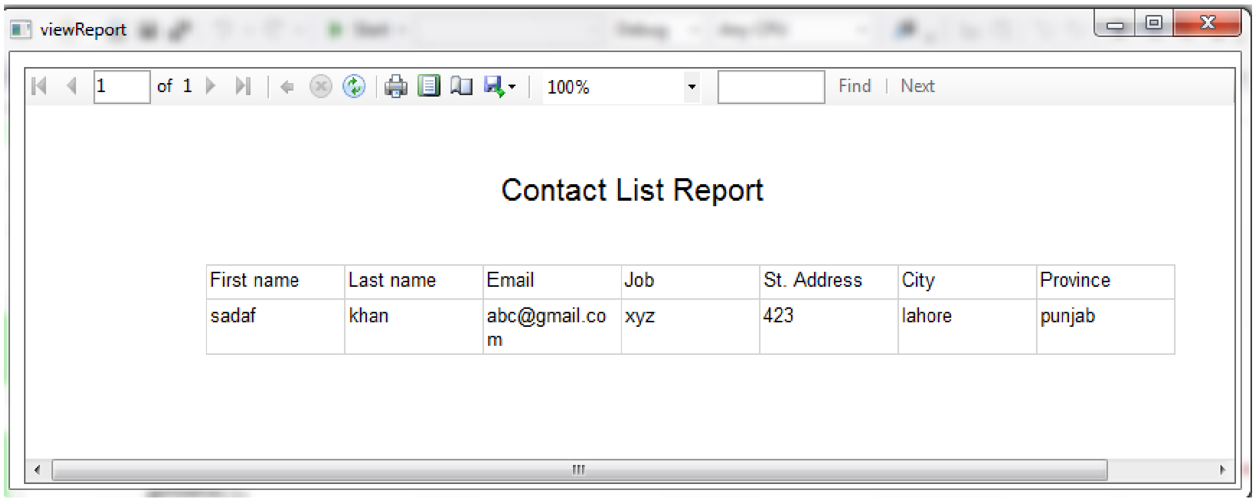


Now back to code behind, write down the following code.

```
    public void viewReport1()
    {
        //Here data table will map the database records to report's data table

        DataTable contact = new DataTable("Contact");
        contact.Columns.Add("fname");
        contact.Columns.Add("lname");
        contact.Columns.Add("email");
        contact.Columns.Add("job");
        contact.Columns.Add("stAddress");
        contact.Columns.Add("city");
        contact.Columns.Add("province");
        foreach (contactList c in list)
        {
          DataRow dr = contact.NewRow();
          dr["fname"] =c.fname;
          dr["lname"] = c.lname;
          dr["email"] = c.email;
          dr["job"] = c.job;
          dr["stAddress"] = c.stAddress;
          dr["city"] = c.city;
          dr["province"] = c.province;
          contact.Rows.Add(dr);
      }
        _reportViewer.LocalReport.DataSources.Clear();  // it will clear the data source of
  //report if any.
        _reportViewer.LocalReport.DataSources.Add(new
  Microsoft.Reporting.WinForms.ReportDataSource("DataSet1", contact));
  // it will set the data source of report. First argument is dataset and 2nd in the table.
        _reportViewer.LocalReport.ReportPath = "../../Report1.rdlc"; // this sets the path of
  //report.
        _reportViewer.LocalReport.EnableExternalImages = true;
        _reportViewer.Visible = true;
```

```
            _reportViewer.RefreshReport();
        }
```



Contact List Report

First name	Last name	Email	Job	St. Address	City	Province
sadaf	khan	abc@gmail.com	xyz	423	lahore	punjab

Task 2:

Make a class **myStack** which inherit **Stack** class. Make two events in this class name as **Pushed** and **Poped**. Then use **event handling** display elements number pushed or poped from stack.
You can use listener class to call push and pop function of your myStack class.
You can **override Push** and **Pop** methods in myStack class.
This is the same task of **myArrayList** as you will see in vedio or in class.

Hint:

```
delegate  void  myHandler(object sender,myEventArg e);

class myStack : Stack
  {
     public event myHandler pushed;
     public event myHandler poped;
     private int c = 1;
     override public void Push(object o)
     {
     }

     void onPushed()
     {
     }
     override public object Pop()
     {
     }
     void onPoped()
     {

     }
```

```
    }
```
Use myStack class in Listner class.

Links:

How to set background images in wpf.

1. https://social.msdn.microsoft.com/Forums/vstudio/en-US/ab245116-547a-451f-a362-97cf17a524cf/how-to-set-background-image-in-the-button-at-runtime-in-wpf?forum=wpf
2. http://stackoverflow.com/questions/3885581/how-to-set-a-background-of-wpf-window
3. http://stackoverflow.com/questions/4009775/change-wpf-window-background-image-in-c-sharp-code