**Lab Assignment 3 – Sarveshwaran Sampathkumar**

1. *Code that uses indexers*

**Model - MemberDB.cs**

class MemberIndexer

{

public ObservableCollection<Member> ocMembers;

/// <summary>

/// Indexer

/// </summary>

/// <param name="i"></param>

/// <returns></returns>

public Member this[int i]

{

get

{

return ocMembers[i];

}

set

{

ocMembers[i] = value;

}

}

}

1. *Code that overloads + operator and – operator. What code do you use to substitue those operator?*

**Model - MemberDB.cs**

/// <summary>

/// Overloaded operator +

/// </summary>

/// <param name="m1"></param>

/// <param name="m"></param>

/// <returns></returns>

public static MemberDB operator +(MemberDB m1, Member m)

{

m1.Add(m);

return m1;

}

/// <summary>

/// Overloaded operator -

/// </summary>

/// <param name="m1"></param>

/// <param name="m"></param>

/// <returns></returns>

public static MemberDB operator -(MemberDB m1, Member m)

{

m1.Remove(m);

return m1;

}

**ViewModel – MainviewModel.cs**

/// <summary>

/// OnMessageReceived Method when receives the MessageMember obj as parameter and then on processing calls the

/// SaveMethod() method.

/// </summary>

/// <param name="obj"></param>

private void OnMessageReceived(MessageMember obj)

{

if (obj.MessageType == ((int)MessageMember.MessageTypeList.AddList))

{

memDB += obj.member;

}

else if (obj.MessageType == ((int)MessageMember.MessageTypeList.UpdateList))

{

if (Members.Contains(obj.memberN))

{

int indexOfMember = Members.IndexOf(obj.memberN);

Members.Remove(obj.memberN);

Members.Insert(indexOfMember, obj.member);

}

}

else if (obj.MessageType == ((int)MessageMember.MessageTypeList.DeleteList))

{

memDB -= obj.memberN;

}

SaveMethod();

}

1. *Code to use the Changed Changed event to save the memberships and refresh the list box any time the list change*

**Model - UpdateViewModel**

/// <summary>

/// UpdateMethod () which take IClosable object as an input.

/// It makes sure that the fields are no empty.

/// If the fields have valid data, then it updates it.

/// </summary>

/// <param name="window"></param>

private void UpdateMethod(ICloseable window)

{

if (EnteredFname == null || EnteredLname == null || EnteredemailId == null)

{

MessageBox.Show("Fields cannot be empty");

}

else

{

MessageMember mmMessage = new MessageMember();

mmMessage.memberN = CurrentMember;

mmMessage.member = new Member(EnteredFname, EnteredLname, EnteredemailId);

mmMessage.MessageType = (int)MessageMember.MessageTypeList.UpdateList;

Messenger.Default.Send<MessageMember>(mmMessage);

window.Close();

}

}

/// <summary>

/// This is a getter and setter for CurrentMemeber property

/// </summary>

public Member CurrentMember

{

get

{

return currMethod;

}

set

{

currMethod = value;

}

}

**ViewModel – MainViewModel.cs**

/// <summary>

/// OnMessageReceived Method when receives the MessageMember obj as parameter and then on processing calls the

/// SaveMethod() method.

/// </summary>

/// <param name="obj"></param>

private void OnMessageReceived(MessageMember obj)

{

if (obj.MessageType == ((int)MessageMember.MessageTypeList.AddList))

{

memDB += obj.member;

}

else if (obj.MessageType == ((int)MessageMember.MessageTypeList.UpdateList))

{

if (Members.Contains(obj.memberN))

{

int indexOfMember = Members.IndexOf(obj.memberN);

Members.Remove(obj.memberN);

Members.Insert(indexOfMember, obj.member);

}

}

else if (obj.MessageType == ((int)MessageMember.MessageTypeList.DeleteList))

{

memDB -= obj.memberN;

}

SaveMethod();

}

**Model – Member.cs**

/// <summary>

/// Getter for Display property

/// </summary>

public string Display

{

get

{

return Firstname + " " + Lastname + " " + Email;

}

}

1. *Code to wire the event handler to the event for the Save.*

**View Model – AddViewModel.cs**

/// <summary>

/// SaveMethod() which takes an object of Window for saving the new data in the listbox.

/// </summary>

/// <param name="window"></param>

public void SaveMethod(Window window)

{

if (window != null)

{

if (EnteredFName != null || EnteredLName != null || EnteredEmailId != null)

{

ChangeHandlerEventArgs args = new ChangeHandlerEventArgs();

mem = new Member(EnteredFName, EnteredLName, EnteredEmailId);

ocMembers.Add(mem);

args.MemberList = ocMembers;

MemberDB memberdb = new MemberDB(ocMembers);

memberdb.OnChanged(args);

window.Close();

}

}

}

/// <summary>

/// This is an eventhandler Event\_OnChanged() to save the data.

/// </summary>

/// <param name="members"></param>

public void Event\_OnChanged(ObservableCollection<Member> members)

{

string Fname="", Lname="", Email="";

foreach (Member m in members)

{

Fname = m.Firstname;

Lname = m.Lastname;

Email = m.Email;

}

mem = new Member(Fname, Lname, Email);

mmMessage = new MessageMember() { member = mem, MessageType = (int)MessageMember.MessageTypeList.AddList };

Messenger.Default.Send<MessageMember>(mmMessage);

}

1. *Code to open the Add form*

**ViewModel – MainviewModel.cs**

/// <summary>

/// This method will show a new window for adding new data.

/// </summary>

public void AddMethod()

{

AddMember amAdd = new AddMember();

amAdd.Show();

}

1. *Code to transfer data from the Add view to the Membership Maintenance view*

**Main View Model**

/// <summary>

/// GetMembership() to retrieve the data from the file.

/// </summary>

/// <returns></returns>

public ObservableCollection<Member> GetMembership()

{

string strRow = "";

string[] strCol;

try

{

if (File.Exists(@"file.txt"))

{

StreamReader srInput = new StreamReader(new FileStream(@"file.txt",FileMode.OpenOrCreate,FileAccess.Read),false);

while(srInput.Peek()!=-1)

{

strRow = srInput.ReadLine();

strCol = strRow.Split('-');

Member m = new Member(strCol[0], strCol[1], strCol[2]);

Members.Add(m);

}

srInput.Close();

}

}

catch(FileNotFoundException)

{

Console.WriteLine("File not found");

}

catch(FormatException)

{

Console.WriteLine("Invalid e-mail address format");

}

return MemberIndexer.ocMembers;

}

/// <summary>

/// SaveMembership method for saving it to the test file.

/// </summary>

/// <param name="memberdata"></param>

public void SaveMembership(ObservableCollection<Member> memberdata)

{

using (StreamWriter swWriter = new StreamWriter(@"file.txt", false))

{

foreach (Member m in memberdata)

{

swWriter.Write(m.Firstname+"-");

swWriter.Write(m.Lastname+"-");

swWriter.Write(m.Email);

swWriter.WriteLine();

}

swWriter.Close();

}

}

**Model – Member.cs**

/// <summary>

/// Getter for Display property

/// </summary>

public string Display

{

get

{

return Firstname + " " + Lastname + " " + Email;

}

}

**View Model – AddViewModel.cs**

public AddViewModel()

{

mmMessage = new MessageMember();

mem = new Member(EnteredFName, ipLName, EnteredEmailId);

ocMembers = new ObservableCollection<Member>();

memDB = new MemberDB(ocMembers);

SaveCommand = new RelayCommand<Window>(SaveMethod);

CancelCommand = new RelayCommand<ICloseable>(CancelMethod);

memDB.Changed += Event\_OnChanged;

}

**View Model – MainViewModel.cs**

/// <summary>

/// OnMessageReceived Method when receives the MessageMember obj as parameter and then on processing calls the

/// SaveMethod() method.

/// </summary>

/// <param name="obj"></param>

private void OnMessageReceived(MessageMember obj)

{

if (obj.MessageType == ((int)MessageMember.MessageTypeList.AddList))

{

memDB += obj.member;

}

else if (obj.MessageType == ((int)MessageMember.MessageTypeList.UpdateList))

{

if (Members.Contains(obj.memberN))

{

int indexOfMember = Members.IndexOf(obj.memberN);

Members.Remove(obj.memberN);

Members.Insert(indexOfMember, obj.member);

}

}

else if (obj.MessageType == ((int)MessageMember.MessageTypeList.DeleteList))

{

memDB -= obj.memberN;

}

SaveMethod();

}