

Lab assignment 4

Model

Member.cs

```
using GalaSoft.MvvmLight;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace GymMembers.Model
{
    /// <summary>
    /// A class that represents a member of a gym.
    /// </summary>
    public class Member : ObservableObject
    {
        /// <summary>
        /// The member's first name.
        /// </summary>
        private string firstName;
        /// <summary>
        /// The member's last name.
        /// </summary>

        public Member() { }

        /// <summary>
        /// Creates a new member.
        /// </summary>
        /// <param name="fName">The member's first name.</param>
        /// <param name="lName">The member's last name.</param>
        /// <param name="mail">The member's e-mail.</param>
        public Member(string fName, string lName, string mail)
        {

        }

        /// <summary>
        /// A property that gets or sets the member's last name, and makes sure it's not
        too long.
        /// </summary>
        /// <returns>The member's last name.</returns>
        public string LastName
        {
            get
            {
                return lastName;
            }
            set
            {
            }
        }
    }
}
```

```

        {
            if (value.Length > TEXT_LIMIT)
            {
                throw new ArgumentException("Too long");
            }

            if (value.Length == 0)
            {
                throw new NullReferenceException();
            }

            lastName = value;
        }
    }

    long.
    /// <summary>
    /// A property that gets or sets the member's e-mail, and makes sure it's not too
    /// </summary>
    /// <returns>The member's e-mail.</returns>
    public string Email
    {
        get
        {
            return email;
        }
        set
        {
            if (value.Length > TEXT_LIMIT)
            {
                throw new ArgumentException("Too long");
            }

            if (value.Length == 0)
            {
                throw new NullReferenceException();
            }

            if (value.IndexOf("@") == -1 || value.IndexOf(".") == -1)
            {
                throw new FormatException();
            }

            email = value;
        }
    }

    /// <summary>
    /// Text to be displayed in the list box.
    /// </summary>
    /// <returns>A concatenation of the member's first name, last name, and e-
    mail.</returns>

    }}

```

MemberDB.cs

```
using GalaSoft.MvvmLight;
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace GymMembers.Model
{
    /// <summary>
    /// A class that uses a text file to store information about the gym members long-
    term.
    /// </summary>
    class MemberDB : ObservableObject
    {
        /// <summary>
        /// The list of members to be saved.
        /// </summary>
        private ObservableCollection<Member> members;

        /// <summary>
        /// Where the database is stored.
        /// </summary>
        private const string filepath = "../members.txt";

        /// <summary>
        /// Creates a new member database.
        /// </summary>
        /// <param name="m">The list to saved from or written to.</param>
        public MemberDB(ObservableCollection<Member> m)
        {
            members = m;
        }

        /// <summary>
        /// Reads the saved text file database into the program's list of members.
        /// </summary>
        /// <returns>The list containing the text file data read in.</returns>
        public ObservableCollection<Member> GetMemberships()
        {
            try
            {
                StreamReader input = new StreamReader(new FileStream(filepath,
                    FileMode.OpenOrCreate, FileAccess.Read));

                input.Close();
            }
            catch (FileNotFoundException)
            {
                Console.WriteLine("File not found");
            }
            catch (FormatException)
            {
            }
        }
    }
}
```

```

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

/// <summary>
/// Saves the program's list of members into the text file database.
/// </summary>
public void SaveMemberships()
{
    StreamWriter output = new StreamWriter(new FileStream(filepath,
        FileMode.Create, FileAccess.Write));

    output.Close();
}
}
}

```

MessageMember.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

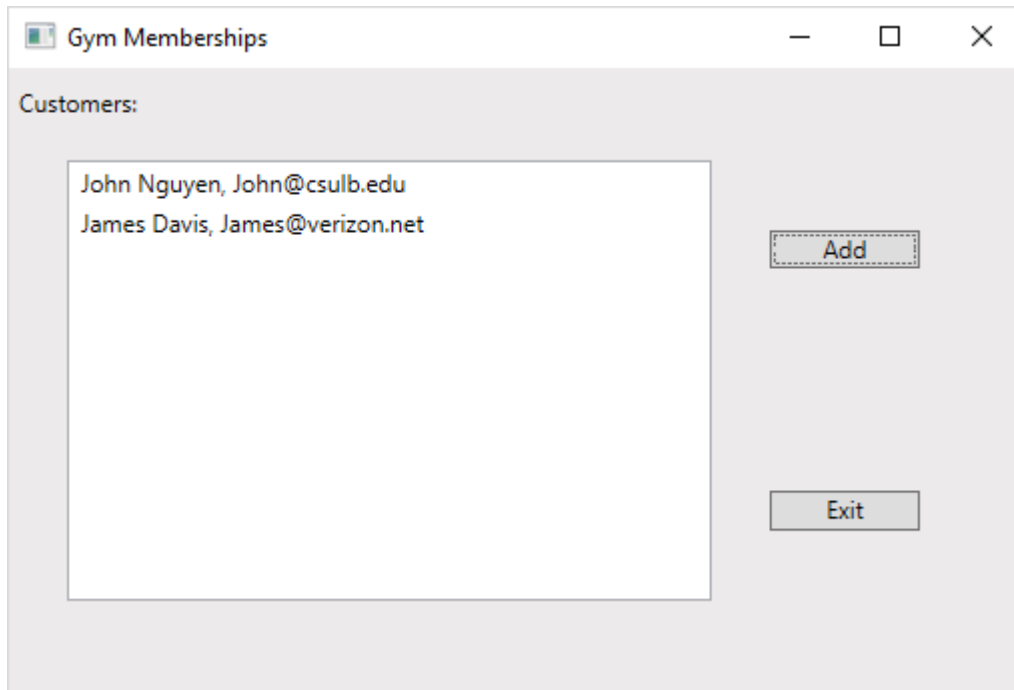
namespace GymMembers.Model
{
    /// <summary>
    /// An extension of member that also includes a message for some sort of extra
    description.
    /// </summary>
    public class MessageMember : Member
    {
        /// <summary>
        /// Creates a new member.
        /// </summary>
        /// <param name="fName">The member's first name.</param>
        /// <param name="lName">The member's last name.</param>
        /// <param name="mail">The member's e-mail.</param>
        /// <param name="message">The extra description</param>
        public MessageMember(string fName, string lName, string mail, string message) :
        base(fName, lName, mail)
        {
            Message = message;
        }

        /// <summary>
        /// A property that includes the message.
        /// </summary>
        public string Message { get; private set; }
    }
}

```

Views

MainWindow.xaml



```
<Window x:Class="GymMembers.View.MainWindow"
        Name="mainWindow"
        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:vm="clr-namespace:GymMembers.ViewModel"
        xmlns:local="clr-namespace:GymMembers"
        xmlns:i="clr-
namespace:System.Windows.Interactivity;assembly=System.Windows.Interactivity"
        mc:Ignorable="d"
        DataContext="{Binding Source={StaticResource Locator}, Path=_____}"
        Title="Gym Memberships" Height="350" Width="525">
    <Grid Background="#FFCEAE">
        <Button x:Name="button" Content="Add" Grid.Row ="1" Grid.Column ="1"
Command="{Binding _____}" HorizontalAlignment="Left" VerticalAlignment="Top"
Width="75" Margin="0,50,0,0"/>
        <Button x:Name="button2" Content="Exit" Grid.Row ="1" Grid.Column ="1"
Command="{Binding ExitCommand}" CommandParameter="{Binding ElementName=mainWindow}"
HorizontalAlignment="Left" VerticalAlignment="Bottom" Width="75" Margin="0,0,0,50"/>
        <ListBox x:Name="listBox" ItemsSource="{Binding _____}"
SelectedItem="{Binding _____}" Grid.Row ="1" Grid.Column ="0"
HorizontalAlignment="Center" VerticalAlignment="Center" Height="220" Width="322">
            <i:Interaction.Triggers>
```

```

        <i:EventTrigger EventName="MouseUp">
            <i:InvokeCommandAction Command="{Binding _____}" />
        </i:EventTrigger>
    </i:Interaction.Triggers>
</ListBox>
<Label x:Name="label" Content="Customers:" Grid.Row ="0" Grid.Column ="0"
HorizontalAlignment="Left" VerticalAlignment="Bottom"/>

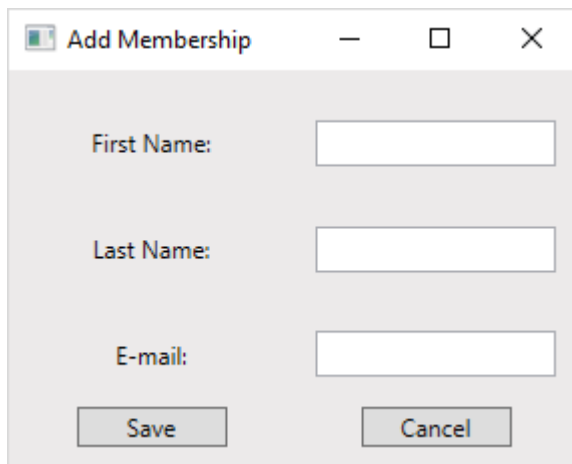
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="380"/>
        <ColumnDefinition />
    </Grid.ColumnDefinitions>

    <Grid.RowDefinitions>
        <RowDefinition Height="*" />
        <RowDefinition Height="250" />
        <RowDefinition Height="*" />
    </Grid.RowDefinitions>

</Grid>
</Window>

```

AddWindow.xaml



```

<Window x:Class="GymMembers.View.AddWindow"
    Name="addWindow"
    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:GymMembers.View"
    mc:Ignorable="d"
    DataContext="{Binding Source={StaticResource Locator}, Path=_____}"
    Title="Add Membership" Height="237" Width="300">
    <Grid Background="#FFECEAEA">
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="*" />

```

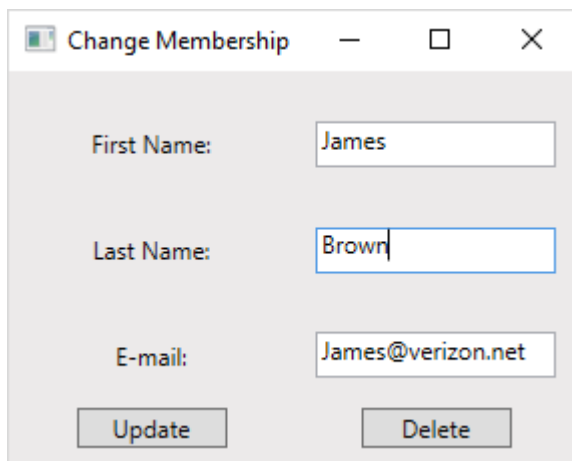
```

        <ColumnDefinition Width="*" />
    </Grid.ColumnDefinitions>
    <Grid.RowDefinitions>
        <RowDefinition Height="10" />
        <RowDefinition Height="*" />
        <RowDefinition Height="*" />
        <RowDefinition Height="*" />
        <RowDefinition Height="30" />
    </Grid.RowDefinitions>
    <Label x:Name="label" Content="First Name:" HorizontalAlignment="Center"
VerticalAlignment="Center" Grid.Row="1" Grid.Column="0" />
    <Label x:Name="label1" Content="Last Name:" HorizontalAlignment="Center"
VerticalAlignment="Center" Grid.Row="2" Grid.Column="0" />
    <Label x:Name="label2" Content="E-mail:" HorizontalAlignment="Center"
VerticalAlignment="Center" Grid.Row="3" Grid.Column="0" />
    <TextBox x:Name="textBox" HorizontalAlignment="Center" Height="23"
TextWrapping="Wrap" VerticalAlignment="Center" Text="{Binding _____}" Width="120"
Grid.Row="1" Grid.Column="1" />
    <TextBox x:Name="textBox1" HorizontalAlignment="Center" Height="23"
TextWrapping="Wrap" VerticalAlignment="Center" Text="{Binding _____}" Width="120"
Grid.Row="2" Grid.Column="1" />
    <TextBox x:Name="textBox2" HorizontalAlignment="Center" Height="23"
TextWrapping="Wrap" VerticalAlignment="Center" Text="{Binding _____}" Width="120"
Grid.Row="3" Grid.Column="1" />
    <Button x:Name="button" Content="Save" HorizontalAlignment="Center"
VerticalAlignment="Top" Command="{Binding _____}" CommandParameter="{Binding
ElementName=_____w}" Width="75" Grid.Row="4" Grid.Column="0" />
    <Button x:Name="button1" Content="Cancel" HorizontalAlignment="Center"
VerticalAlignment="Top" Command="{Binding _____}" CommandParameter="{Binding
ElementName=_____--}" Width="75" Grid.Row="4" Grid.Column="1" />

</Grid>
</Window>

```

ChangeWindow.axml



The screenshot shows a window titled "Change Membership" with a standard Windows title bar (minimize, maximize, close buttons). The window contains three text input fields with labels to their left: "First Name:" with the value "James", "Last Name:" with the value "Brown", and "E-mail:" with the value "James@verizon.net". Below these fields are two buttons: "Update" and "Delete".

App.xaml

```
<Application x:Class="GymMembers.App"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:local="clr-
namespace:GymMembers" StartupUri="View/MainWindow.xaml"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008" d1p1:Ignorable="d"
xmlns:d1p1="http://schemas.openxmlformats.org/markup-compatibility/2006">
    <Application.Resources>
        <ResourceDictionary>
            <vm:ViewModelLocator x:Key="Locator" d:IsDataSource="True" xmlns:vm="clr-
namespace:GymMembers.ViewModel" />
        </ResourceDictionary>
    </Application.Resources>
</Application>
```

ViewModel

MainViewModel.cs

```
using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Command;
using GalaSoft.MvvmLight.Messaging;
using GymMembers.Model;
using GymMembers.View;
using System;
using System.Collections.ObjectModel;
using System.IO;
using System.Windows;
using System.Windows.Input;

namespace GymMembers.ViewModel
{
    /// <summary>
    /// The VM for the main screen that shows the member list.
    /// </summary>
    public class MainViewModel : ViewModelBase
    {
        /// <summary>
        /// The list of registered members.
        /// </summary>
        private ObservableCollection<Member> members;

        /// <summary>
        /// The currently selected member.
        /// </summary>
        private Member selectedMember;

        /// <summary>
        /// The database that keeps track of saving and reading the registered members.
    }
}
```



```

/// </summary>
private MemberDB database;

/// <summary>
/// Initializes a new instance of the MainViewModel class.
/// </summary>
public MainViewModel()
{
    members =
    database =
    members = database.GetMemberships();
    AddCommand =
    ExitCommand =
    ChangeCommand =
    Messenger.Default.Register<MessageMember>(this, ReceiveMember);
    Messenger.Default.Register<NotificationMessage>(this, ReceiveMessage);
}

/// <summary>
/// The command that triggers adding a new member.
/// </summary>
public ICommand AddCommand { get; private set; }

/// <summary>
/// The currently selected member in the list box.
/// </summary>
public Member SelectedMember
{
    get
    {
        return selectedMember;
    }
    set
    {
        selectedMember = value;
        RaisePropertyChanged("SelectedMember");
    }
}

/// <summary>
/// Shows a new add screen.
/// </summary>
public void AddMethod()
{
    AddWindow add = new AddWindow();
    add.Show();
}

/// <summary>
/// Closes the application.
/// </summary>
/// <param name="window">The window to close.</param>
public void ExitMethod(IClosable window)
{
    if (window != null)
    {

```

```

        window.Close();
    }
}

/// <summary>
/// Opens the change window.
/// </summary>
public void ChangeMethod()
{
    if (SelectedMember != null)
    {
        ChangeWindow change = new ChangeWindow();
        change.Show();
        Messenger.Default.Send(_____);
    }
}

/// <summary>
/// Gets a new member for the list.
/// </summary>
/// <param name="m">The member to add. The message denotes how it is added.
/// "Update" replaces at the specified index, "Add" adds it to the list.</param>
public void ReceiveMember(MessageMember m)
{
    if (m.Message == "Update")
    {
        _____
        database.SaveMemberships();
    }
    else if (m.Message == "Add")
    {
        _____
        database.SaveMemberships();
    }
}

/// <summary>
/// Gets text messages.
/// </summary>
/// <param name="msg">The received message. "Delete" means the currently selected
member is deleted.</param>
public void ReceiveMessage(NotificationMessage msg)
{
    if (msg.Notification == "Delete")
    {
        _____
        database.SaveMemberships();
    }
}

/// <summary>
/// The list of registered members.
/// </summary>
public ObservableCollection<Member> MemberList
{
    get { return members; }
}

```

```

    }
}

```

AddViewModel.cs

```

using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Command;
using GalaSoft.MvvmLight.Messaging;
using GymMembers.Model;
using System;
using System.Collections.ObjectModel;
using System.IO;
using System.Windows;
using System.Windows.Input;

namespace GymMembers.ViewModel
{
    /// <summary>
    /// The VM for adding users to the list.
    /// </summary>
    public class AddViewModel : ViewModelBase
    {
        /// <summary>
        /// The currently entered first name in the add window.
        /// </summary>
        private string enteredFName;

        /// <summary>
        /// The currently entered last name in the add window.
        /// </summary>
        private string enteredLName;

        /// <summary>
        /// The currently entered email in the add window.
        /// </summary>
        private string enteredEmail;

        /// <summary>
        /// Initializes a new instance of the AddViewModel class.
        /// </summary>
        public AddViewModel()
        {
            SaveCommand = new RelayCommand<IClosable>(SaveMethod);
        }

        /// <summary>
        /// The command that triggers saving the filled out member data.
        /// </summary>
        public ICommand SaveCommand { get; private set; }

        /// <summary>
        /// The command that triggers closing the add window.
        /// </summary>
        public ICommand CancelCommand { get; private set; }
    }
}

```

```

    /// <summary>
    /// Sends a valid member to the Main VM to add to the list, then closes the
window.
    /// </summary>
    /// <param name="window">The window to close.</param>
    public void SaveMethod(IClosable window)
    {
        try
        {
            if (window != null)
            {
                Messenger.Default.Send(_____);
                window.Close();
            }
        }
        catch (ArgumentException)
        {
            MessageBox.Show("Fields must be under 25 characters.", "Entry Error");
        }
        catch (NullReferenceException)
        {
            MessageBox.Show("Fields cannot be empty.", "Entry Error");
        }
        catch (FormatException)
        {
            MessageBox.Show("Must be a valid e-mail address.", "Entry Error");
        }
    }

    /// <summary>
    /// Closes the window.
    /// </summary>
    /// <param name="window">The window to close.</param>
    public void CancelMethod(IClosable window)
    {
        if (window != null)
        {
            window.Close();
        }
    }

    /// <summary>
    /// The currently entered first name in the add window.
    /// </summary>
    public string EnteredFName
    {
        get
        {
            return enteredFName;
        }
        set
        {
            enteredFName = value;
            RaisePropertyChanged("EnteredFName");
        }
    }

    /// <summary>

```

```
    }
}
```

ChangeViewModel.cs

```
using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Command;
using GalaSoft.MvvmLight.Messaging;
using GymMembers.Model;
using System;
using System.Collections.ObjectModel;
using System.IO;
using System.Windows;
using System.Windows.Input;

namespace GymMembers.ViewModel
{
    /// <summary>
    /// The VM for modifying or removing users.
    /// </summary>
    public class ChangeViewModel : ViewModelBase
    {
        /// <summary>
        /// The currently entered first name in the change window.
        /// </summary>
        private string enteredFName;

        /// <summary>
        /// The currently entered last name in the change window.
        /// </summary>
        private string enteredLName;

        /// <summary>
        /// The currently entered email in the change window.
        /// </summary>
        private string enteredEmail;

        /// <summary>
        /// Initializes a new instance of the ChangeViewModel class.
        /// </summary>
        public ChangeViewModel()
        {
            Messenger.Default.Register<Member>(this, _____-);
        }

        /// <summary>
        /// The command that triggers saving the filled out member data.
        /// </summary>
        public ICommand UpdateCommand { get; private set; }

        /// <summary>
        /// The command that triggers removing the previously selected user.
        /// </summary>
        public ICommand DeleteCommand { get; private set; }

        /// <summary>
```

```

    /// Sends a valid member to the main VM to replace at the selected index with,
    then closes the change window.
    /// </summary>
    /// <param name="window">The window to close.</param>
    public void UpdateMethod(IClosable window)
    {
        try
        {
            Messenger.Default.Send(_____--));
            window.Close();
        }
        catch (ArgumentException)
        {
            MessageBox.Show("Fields must be under 25 characters.", "Entry Error");
        }
        catch (_____n)
        {
            MessageBox.Show("Fields cannot be empty.", "Entry Error");
        }
        catch (_____n)
        {
            MessageBox.Show("Must be a valid e-mail address.", "Entry Error");
        }
    }

    /// <summary>
    /// Sends out a message to initiate closing the change window.
    /// </summary>
    /// <param name="window">The window to close.</param>
    public void DeleteMethod(IClosable window)
    {
        if (window != null)
        {
            Messenger.Default.Send(_____--));
            window.Close();
        }
    }

    /// <summary>
    /// Receives a member from the main VM to auto-fill the change box with the
    currently selected member.
    /// </summary>
    /// <param name="m">The member data to fill in.</param>
    public void GetSelected(Member m)
    {
        _____
    }

    /// <summary>
    /// The currently entered first name in the change window.
    /// </summary>
    public string EnteredFName
    {
        get
        {
            return enteredFName;
        }
        set

```

```

        {
            enteredFName = value;
            RaisePropertyChanged("EnteredFName");
        }
    }
}

```

ViewModelLocator.cs

```

using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Ioc;
using Microsoft.Practices.ServiceLocation;
using System;

namespace GymMembers.ViewModel
{
    /// <summary>
    /// This class contains static references to all the view models in the
    /// application and provides an entry point for the bindings.
    /// </summary>
    public class ViewModelLocator
    {
        /// <summary>
        /// Initializes a new instance of the ViewModelLocator class.
        /// </summary>
        public ViewModelLocator()
        {
            ServiceLocator.SetLocatorProvider(() => SimpleIoc.Default);
            SimpleIoc.Default.Register<MainViewModel>();
        }

        /// <summary>
        /// A property that lets the main window connect with its View Model.
        /// </summary>
        public MainViewModel Main
        {
            get
            {
                return ServiceLocator.Current.GetInstance<MainViewModel>();
            }
        }
    }
}

```

IClosable.cs

```

namespace GymMembers
{
    /// <summary>
    /// An interface that lets objects be closed.
    /// </summary>

```

```
public interface IClosable
{
    /// <summary>
    /// Closes this object.
    /// </summary>
    void Close();
}
}
```