НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КПІ ім. Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра програмного забезпечення комп’ютерних систем

ЛАБОРАТОРНА РОБОТА №5

з дисципліни

«Об’єктно-орієнтоване програмування»

ТЕМА: «C# .Net. Розробка програмного забезпечення з GUI»

Підготував: студент групи КП-52

Сорочинський Владислав

Перевірила:

Заболотня Тетяна Миколаївна

Київ 2016

**Мета роботи**

Ознайомитися з шаблонами для створення додатків з графічним інтерфейсом (GUI).

**Постановка завдання**

1. На основі шаблону MVVM розробити програмне забезпечення з графічним інтерфейсом користувача для роботи з даними, які містяться у вже існуючих класах. Для цього використати шаблон проекту типа WPF.

2. В розроблюваному програмному забезпеченні передбачити такі функціональні можливості:

- проглядання даних колекції (readonly);

- додавання даних до колекції (з перевіркою їх коректності: у разі виявлення некоректних даних деактивувати кнопку «Додати»);

- редагування елементів колекції;

- видалення елементів з колекції (без заповнення додаткових полів; кнопку «Видалити» розмістити в кожному рядку таблиці);

- зберігання поточного елементу колекції в окремому полі VM.

3. Забезпечити вирівнювання елементів керування в GUI по горизонталі та вертикалі

4. Всі контейнери даних (Grid, Textbox тощо) повинні бути підписані у відповідних Label-ах.

5. Завантаження даних до колекції має бути з файла довільного формату під час старту програми (десеріалізація). Запис змінених даних до файлу відбувається під час завершення роботи програми (серіалізація).

6. При розробленні дотримуватись правил написання чистого коду.

**Лістинг програми**

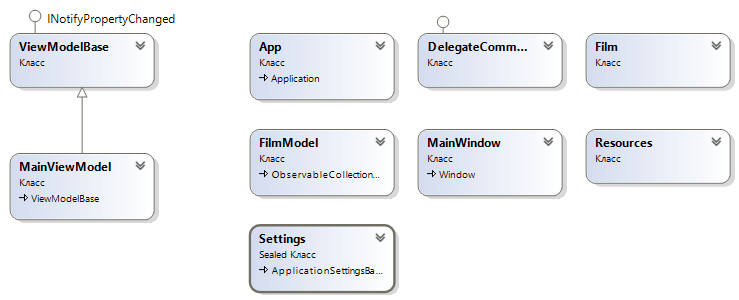
|  |
| --- |
| FilmModel.cs |
| using System;  using System.Collections.Generic;  using System.Collections.ObjectModel;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Runtime.Serialization;  using System.Runtime.Serialization.Json;  using System.IO;  namespace lab5  {  [DataContract]  public class Film  {  [DataMember]  public string Name { get; set; }  [DataMember]  public int Year {get; set; }  [DataMember]  public string Country {get; set; }  [DataMember]  public double Rating { get; set; }  public Film(string name, int year, string country, double rating)  {  this.Name = name;  this.Year = year;  this.Country = country;  this.Rating = rating;  }  }  public class FilmModel: ObservableCollection<Film>  {  private static object \_threadLock = new Object();  private static FilmModel current = null;  public static FilmModel Current  {  get  {  lock (\_threadLock)  if (current == null)  current = new FilmModel();  return current;  }  }  private FilmModel()  {  DataContractJsonSerializer jsonFormatter = new DataContractJsonSerializer(typeof(Film[]));  using (FileStream fs = new FileStream("films.json", FileMode.OpenOrCreate))  {  Film[] films = (Film[])jsonFormatter.ReadObject(fs);  foreach (Film f in films)  {  Add(f);  }  }  }  public void AddAFilm(string name, int year, string country, double rating)  {  Film aNewFilm = new Film(name, year,  country, rating);  Add(aNewFilm);  }  }  } |

|  |
| --- |
| MainViewModel.cs |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows;  using System.Windows.Input;  using System.Runtime.Serialization;  using System.Runtime.Serialization.Json;  using System.IO;  namespace lab5  {  public class ViewModelBase : INotifyPropertyChanged  {  #region Implementation of INotifyPropertyChanged  public event PropertyChangedEventHandler PropertyChanged;  protected virtual void OnPropertyChanged(string propertyName)  {  OnPropertyChanged(new PropertyChangedEventArgs(propertyName));  }  protected virtual void OnPropertyChanged(PropertyChangedEventArgs args)  {  var handler = PropertyChanged;  if (handler != null)  handler(this, args);  }  #endregion  }  public class DelegateCommand : ICommand  {  public delegate void ICommandOnExecute(object parameter);  public delegate bool ICommandOnCanExecute(object parameter);  private ICommandOnExecute \_execute;  private ICommandOnCanExecute \_canExecute;  public DelegateCommand(ICommandOnExecute onExecuteMethod, ICommandOnCanExecute onCanExecuteMethod)  {  \_execute = onExecuteMethod;  \_canExecute = onCanExecuteMethod;  }  #region ICommand Members  public event EventHandler CanExecuteChanged  {  add { CommandManager.RequerySuggested += value; }  remove { CommandManager.RequerySuggested -= value; }  }  public bool CanExecute(object parameter)  {  return \_canExecute.Invoke(parameter);  }  public void Execute(object parameter)  {  \_execute.Invoke(parameter);  }  #endregion  }  public class MainViewModel : ViewModelBase  {  private DelegateCommand exitCommand;  #region Constructor  public FilmModel Films { get; set; }  public string FilmNameToAdd { get; set; }  public int FilmYearToAdd { get; set; }  public string FilmCountryToAdd { get; set; }  public double FilmRatingToAdd { get; set; }  public MainViewModel()  {  Films = FilmModel.Current;  }  #endregion  public ICommand ExitCommand  {  get  {  if (exitCommand == null)  {  exitCommand = new DelegateCommand(Exit, CanExecuteCommand1);  }  return exitCommand;  }  }  private void Exit(object parameter)  {  DataContractJsonSerializer jsonFormatter = new DataContractJsonSerializer(typeof(Film[]));  using (FileStream fs = new FileStream("films.json", FileMode.OpenOrCreate))  {  Film[] filmArray = Films.ToArray();  jsonFormatter.WriteObject(fs, filmArray);  }  Application.Current.Shutdown();  }  public bool CanExecuteCommand1(object parameter)  {  return true;  }  private ICommand \_AddFilm;  public ICommand AddFilm  {  get  {  if (\_AddFilm == null)  {  \_AddFilm = new DelegateCommand(ExecuteCommand1, CanExecuteCommand1);  }  return \_AddFilm;  }  }  public void ExecuteCommand1(object parameter)  {  FilmNameToAdd.Trim();  StringBuilder SB = new StringBuilder();  bool valid = true;  if (FilmNameToAdd == "")  {  SB.Remove(0, SB.Length);  SB.Append("Please type in a name for the film.");  valid = false;  //throw new ArgumentException(SB.ToString());  }  if ((FilmYearToAdd < 1895) || (FilmYearToAdd > 2017))  {  SB.Remove(0, SB.Length);  SB.Append("Film year of release has to be " +  "between 1895 and 2017. ");  SB.Append("Please give a enter year");  valid = false;  //throw new ArgumentException(SB.ToString());  }  if ((FilmRatingToAdd < 0) || (FilmRatingToAdd > 10))  {  SB.Remove(0, SB.Length);  SB.Append("Film rating has to be " +  "between 0 and 10 (could be double)");  valid = false;  //throw new ArgumentException(SB.ToString());  }  if (valid)  Films.AddAFilm (FilmNameToAdd,  FilmYearToAdd, FilmCountryToAdd, FilmRatingToAdd);  }  }  } |

|  |
| --- |
| MainWindow.xaml |
| <Window x:Class="lab5.MainWindow"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  Title="MainWindow" Height="350" Width="525">  <Window.Resources>  <!--<c:CommandReference x:Key="ExitCommandReference"  Command="{Binding ExitCommand}" />-->  <Style x:Key="LabelStyle" TargetType="{x:Type TextBlock}">  <Setter Property="FontWeight" Value="Bold" />  </Style>  <Style x:Key="GridViewHeaderStyle"  TargetType="{x:Type GridViewColumnHeader}">  <Setter Property="FontWeight" Value="Bold" />  <Setter Property="Foreground" Value="Maroon" />  <Setter Property="Background" Value="LightSkyBlue" />  </Style>  </Window.Resources>  <!--<Window.InputBindings>  <KeyBinding Key="X" Modifiers="Control"  Command="{StaticResource ExitCommandReference}" />  </Window.InputBindings>-->  <Grid>  <Grid.RowDefinitions>  <RowDefinition Height="Auto" />  <RowDefinition Height="40" />  <RowDefinition Height="\*" />  </Grid.RowDefinitions>  <Menu Grid.Row="0">  <MenuItem Header="\_File">  <MenuItem Command="{Binding ExitCommand}"  Header="E\_xit" InputGestureText="Ctrl-X" />  </MenuItem>  </Menu>  <Grid Grid.Row="1" HorizontalAlignment="Right"  Margin="0, 5, 20, 10" VerticalAlignment="Center">  <StackPanel Orientation="Horizontal">  <TextBlock Style="{StaticResource LabelStyle}">  Film name</TextBlock>  <TextBox Width="71" Margin="10, 0, 5, 0"  Text="{Binding Path=FilmNameToAdd,  Mode=OneWayToSource}">  </TextBox>  <TextBlock Style="{StaticResource LabelStyle}" Text="Year"/>  <TextBox Width="45" Margin="10, 0, 5, 0"  Text="{Binding Path=FilmYearToAdd,  Mode=OneWayToSource}">  </TextBox>  <TextBlock Style="{StaticResource LabelStyle}" Text="Country"/>  <TextBox Width="45" Margin="10, 0, 5, 0"  Text="{Binding Path=FilmCountryToAdd,  Mode=OneWayToSource}">  </TextBox>  <TextBlock Style="{StaticResource LabelStyle}" Text="Rating"/>  <TextBox Width="45" Margin="10, 0, 5, 0"  Text="{Binding Path=FilmRatingToAdd,  Mode=OneWayToSource}">  </TextBox>  <Button x:Name="btnAddFilm"  Content="Add a film"  Command="{Binding AddFilm}" Click="btnAddFilm\_Click">  </Button>  </StackPanel>  </Grid>  <ListView Grid.Row="2" BorderBrush="White"  ItemsSource="{Binding Path=Films}"  HorizontalAlignment="Stretch" SelectionChanged="ListView\_SelectionChanged">  <ListView.View>  <GridView>  <GridViewColumn Header="Name"  HeaderContainerStyle=  "{StaticResource GridViewHeaderStyle}"  DisplayMemberBinding="{Binding Path=Name}" />  <GridViewColumn Header="Year"  HeaderContainerStyle=  "{StaticResource GridViewHeaderStyle}"  DisplayMemberBinding="{Binding Path=Year}" />  <GridViewColumn Header="Country"  HeaderContainerStyle=  "{StaticResource GridViewHeaderStyle}"  DisplayMemberBinding="{Binding Path=Country}" />  <GridViewColumn Header="Rating"  HeaderContainerStyle=  "{StaticResource GridViewHeaderStyle}"  DisplayMemberBinding="{Binding Path=Rating}" />    </GridView>  </ListView.View>  </ListView >  </Grid>  </Window> |

|  |
| --- |
| MainWindow.xaml.cs |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows;  using System.Windows.Controls;  using System.Windows.Data;  using System.Windows.Documents;  using System.Windows.Input;  using System.Windows.Media;  using System.Windows.Media.Imaging;  using System.Windows.Navigation;  using System.Windows.Shapes;  namespace lab5  {  /// <summary>  /// Логика взаимодействия для MainWindow.xaml  /// </summary>  public partial class MainWindow : Window  {  public MainWindow()  {  InitializeComponent();  DataContext = new MainViewModel();  }  private void ListView\_SelectionChanged(object sender, SelectionChangedEventArgs e)  {  }  private void btnAddFilm\_Click(object sender, RoutedEventArgs e)  {  }  }  } |

**Діаграма класів**

****

**Висновки**

В ході виконання лабораторної роботи я ознайомився з шаблонами для створення додатків з графічним інтерфейсом (GUI).