МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

ФГБОУ ВО «БАЙКАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Кафедра математических методов и цифровых технологий

**Разработка приложений для мобильных устройств**

Отчёт по второй лабораторной работе. Вариант 14.

|  |  |  |
| --- | --- | --- |
| студента бакалавриата группы | БИ-17-1 | Точилова Е.А. Фамилия И.О. |
| проверил | к.т.н., доцент кафедры математических методов и цифровых технологий  ученое звание, должность | Родионов А.В.  Фамилия И.О |

Иркутск, 2020

# Windows.Storage.StorageFile

Класс **Windows.Storage.StorageFile** представляет файл. Этот класс содержит свойства, которые предоставляют сведения о файле, и методы для создания, открытия, копирования, удаления и переименования файлов. Возможно, вы привыкли работать со строковыми путями. Есть ряд API среды выполнения Windows, принимающих строковый путь, но чаще всего вы будете использовать объект StorageFile для представления файла, так как некоторые файлы на платформе UWP могут быть без пути или путь к ним может быть слишком длинным. Используйте метод StorageFile.GetFileFromPathAsync() для преобразования строкового пути в объект StorageFile.

# Windows.Storage.FileIO

Класс **Windows.Storage.FileIO** предоставляет простой способ для чтения и записи текста. Этот класс также может читать и записывать массив байтов или содержимое буфера. Этот класс очень похож на класс PathIO. Основное различие заключается в том, что вместо использования строкового пути, как в PathIO, он принимает StorageFile.

# Windows.Storage.StorageFolder

Класс **Windows.Storage.StorageFolder** представляет папку (каталог).

Этот класс содержит методы для создания файлов, запроса содержимого папки, создания, переименования и удаление папок, а также свойства, которые предоставляют сведения о папке.

# Манифнифест приложения

Манифест пакета — это XML-документ, содержащий сведения, необходимые системе для развертывания, вывода или обновления приложения Windows. Эта информация включает в себя идентификатор пакета, зависимости пакета, требуемые возможности, визуальные элементы и точки расширения. Каждый пакет приложения должен содержать один манифест пакета.

Манифест пакета имеет цифровую подпись в процессе подписи пакета приложения. После подписания манифест нельзя изменить без недействительной подписи пакета. После установки пакета файл манифеста пакета появится в каталоге установленного пакета.

# Windows.Storage.KnownFolders

Кроме описанных в прошлой теме папок, которые доступны разработчику, мы также можем использовать еще ряд папок, которые являются специальными и предназначены для хранения файлов пользователя. Доступ к ним можно получить через свойства класса Windows.Storage.KnownFolders:

DocumentsLibrary: представляет папку Documents (Документы)

HomeGroup: каталог HomeGroup

MediaServerDevices: папка Media Server Devices

MusicLibrary: папка Music (Музыка)

PicturesLibrary: папка Pictures (Изображения)

RemovableDevices: папка Removable Devices

VideosLibrary: папка Videos (Видео)

SavedPictures: папка для сохраненных скриншотов

# Windows.Storage.Pickers.FileSavePicker

На Universal Windows Platform для работы с файлами мы можем использовать специальные диалоговые окна. В каких ситуациях они могут быть нам полезны? Прежде всего когда речь идет об открытии или сохранении файла, и нам нужен простой и понятный визуальный компонент для выбора места файла на устройстве.

Для сохранения файла применяется компонент Windows.Storage.Pickers.FileSavePicker, а для открытия файла .

# System.Runtime.Serialization

Содержит классы, которые можно использовать для сериализации и десериализации объектов. Сериализацией называется процесс преобразования объекта или набора объектов в линейную последовательность байтов для сохранения или для передачи на другой компьютер. Десериализация — это процесс получения сохраненных данных и восстановления из них объектов.

# SQLite и Entity Framework

SQLite являются официально рекомендуемой системой управления баз данных для создания локального хранилища данных в Windows 10.

Для работы с SQLite в приложении на UWP мы можем использовать разные подходы: либо работать с SQLite через одну из библиотек, которых довольно много в репозитории NuGet, либо использовать Entity Framework Core, который абстрагирует от структуры базы данных и позволяет работать с данными как с объектами стандартных классов языка C#. Наиболее простым и рекомендуемым подходом является использование Entity Framework.

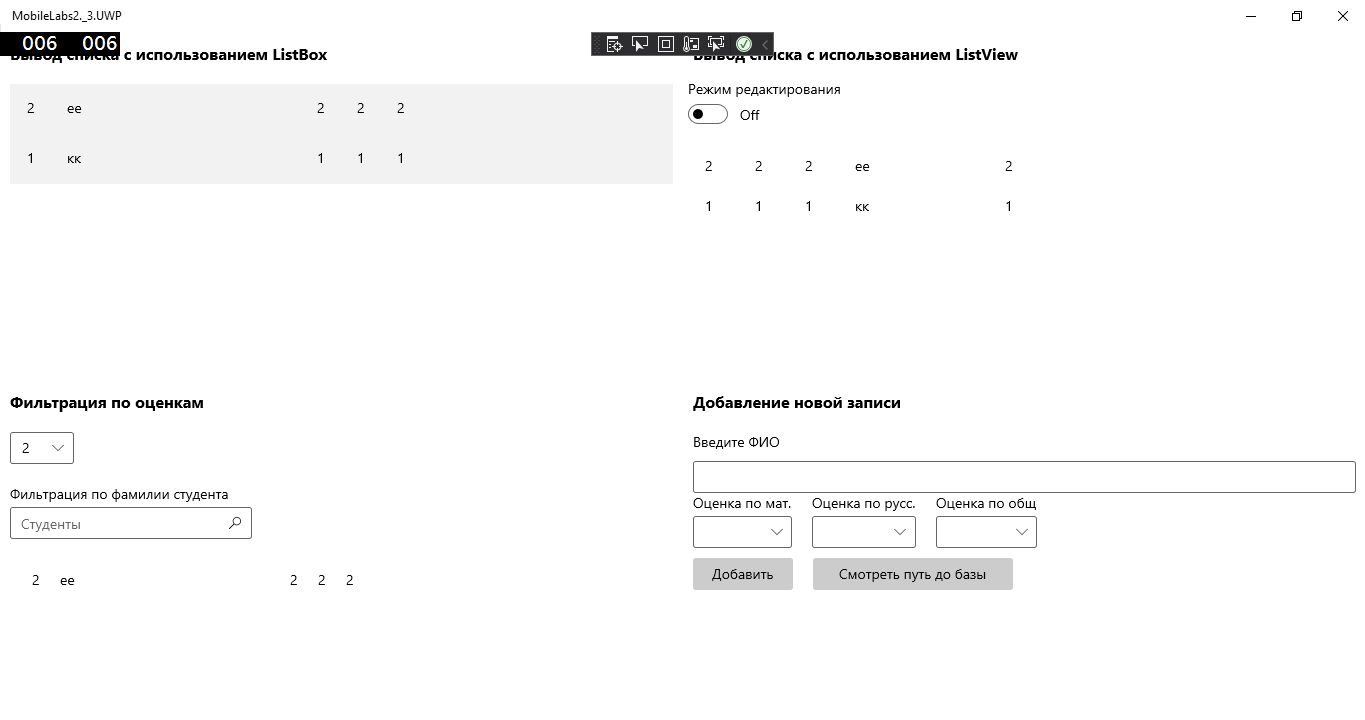
**Задание лабораторной работы**

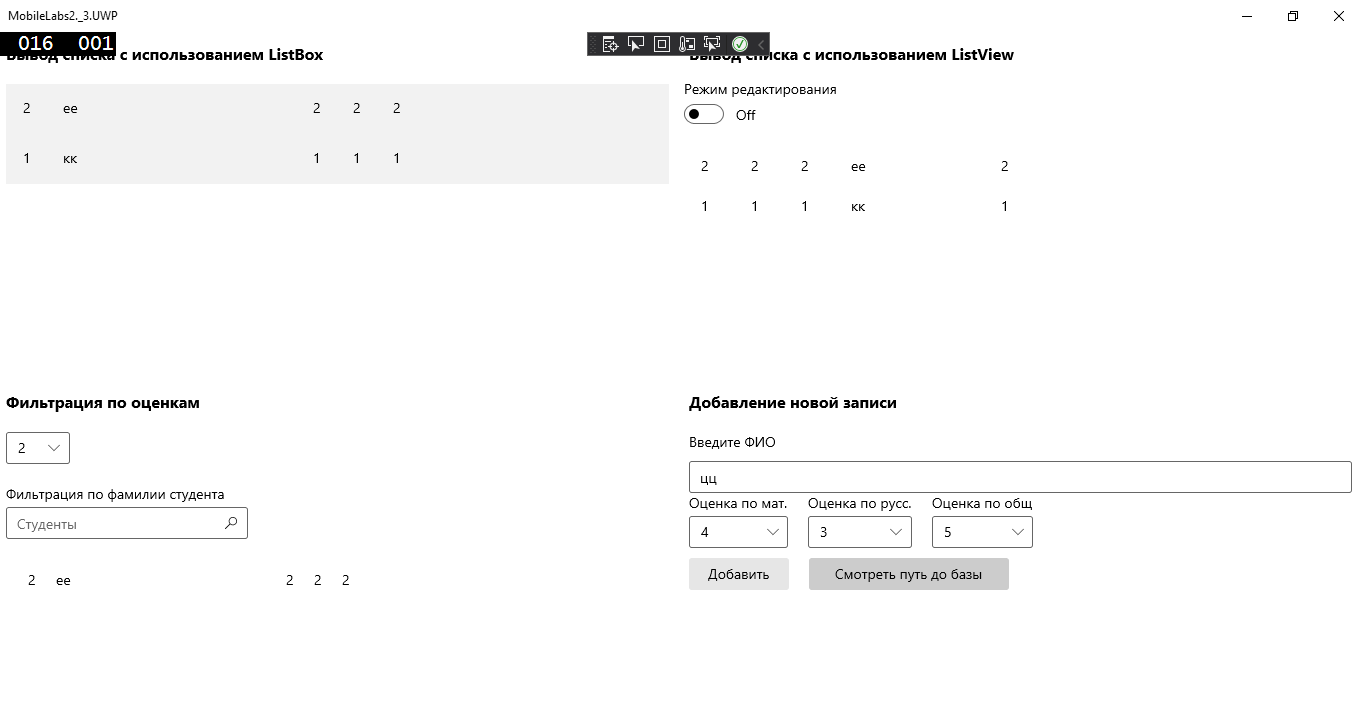
Для задания из прошлой части лабораторной работы реализовать функции сохранения данных в файл или базу данных. 2. Ответьте на вопросы преподавателя, при необходимости выполните дополнительные задания, которые даст преподаватель.

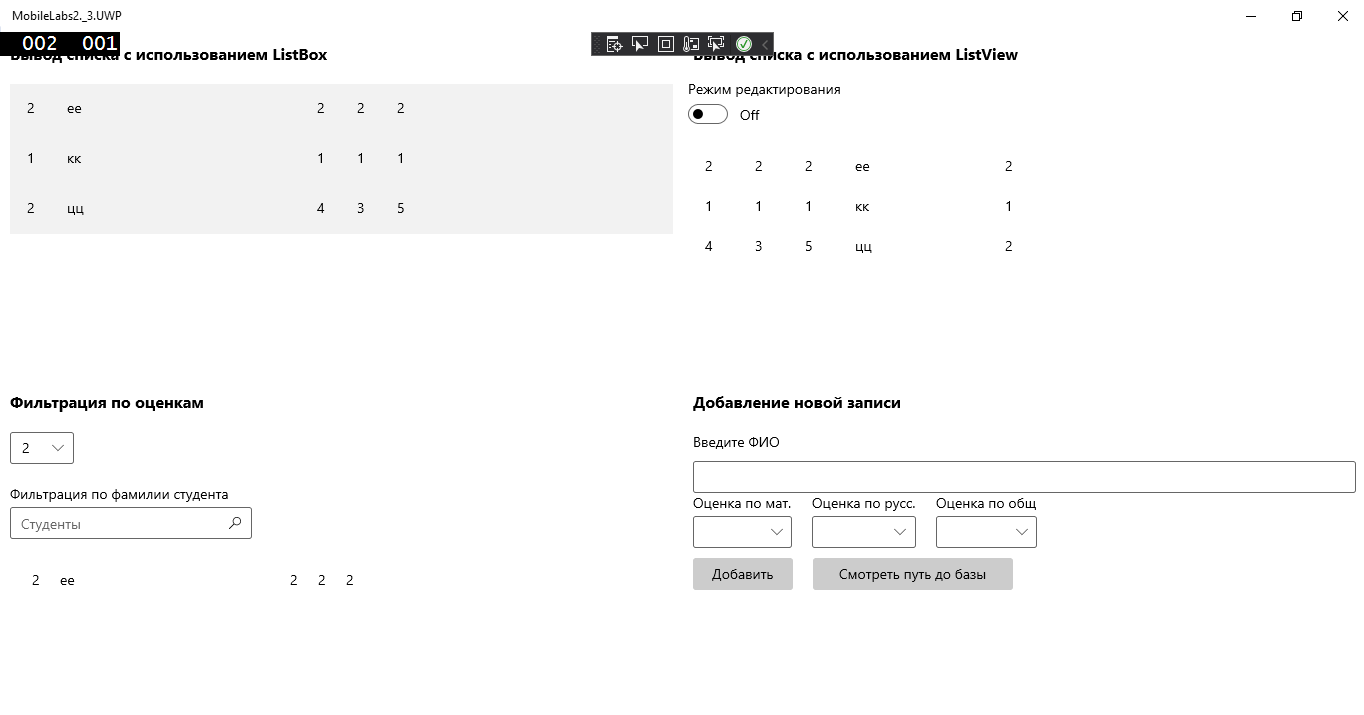
Вариант: 14

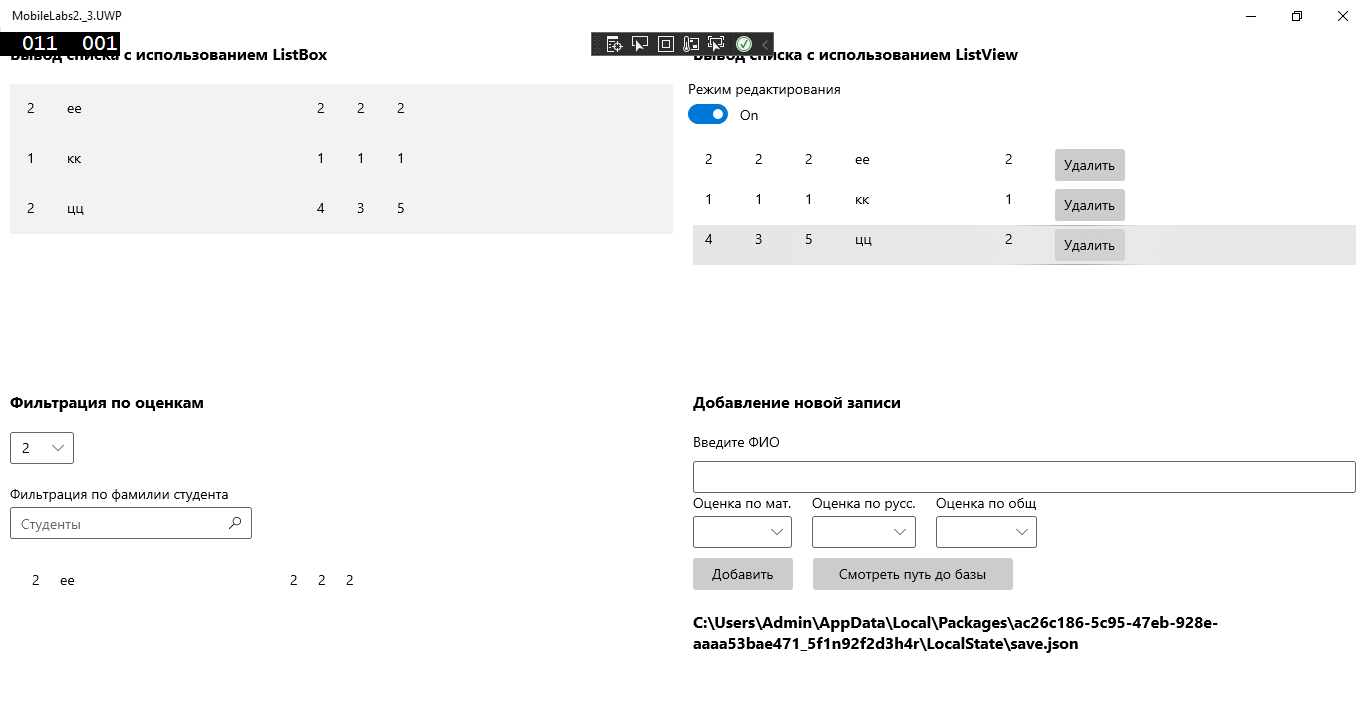
Ведомость абитуриентов, сдавших вступительные экзамены в университет, содержит: Ф.И.О. абитуриента, оценки. Определить средний балл по университету и вывести список абитуриентов, средний балл которых выше среднего балла по университету. Первыми в списке должны идти студенты, сдавшие все экзамены на 5.

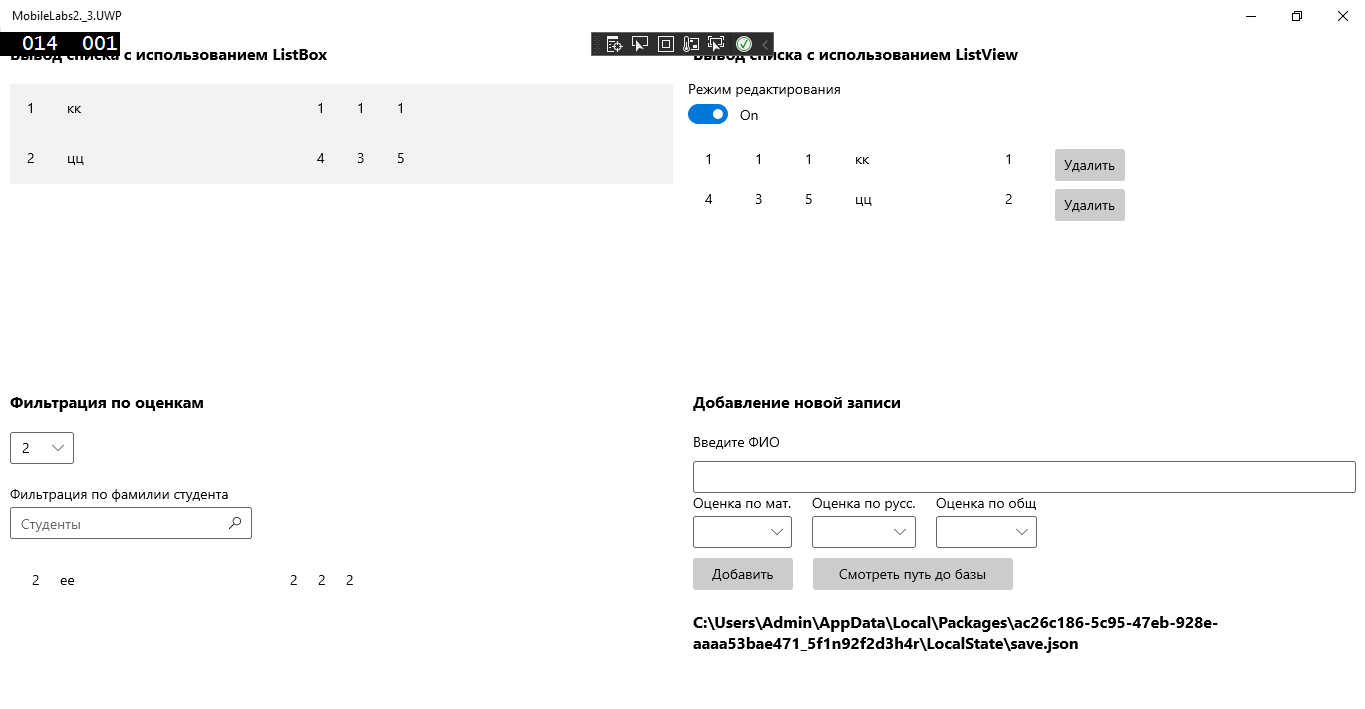
**Скриншоты лабораторной работы**











**Код программы MainPage.xaml.cs**

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices.WindowsRuntime;

using Windows.Foundation;

using Windows.Foundation.Collections;

using Windows.UI.Xaml;

using Windows.UI.Xaml.Controls;

using Windows.UI.Xaml.Controls.Primitives;

using Windows.UI.Xaml.Data;

using Windows.UI.Xaml.Input;

using Windows.UI.Xaml.Media;

using Windows.UI.Xaml.Navigation;

namespace MobileLabs2.\_3.UWP

{

public sealed partial class MainPage

{

//Вариант: №14

//Ведомость абитуриентов, сдавших вступительные экзамены в университет, содержит:

//Ф.И.О.абитуриента, оценки.Определить средний балл по университету и вывести список

//абитуриентов, средний балл которых выше среднего балла по университету.Первыми в списке

//должны идти студенты, сдавшие все экзамены на 5

//public ObservableCollection<Student> allStudents { get; set; }

allStudents ast = new allStudents();

public MainPage()

{

this.InitializeComponent();

try

{

ast.openFile();

}

catch

{

ast.saveFile();

}

}

private void suggestBox\_QuerySubmitted(AutoSuggestBox sender, AutoSuggestBoxQuerySubmittedEventArgs args)

{

var term = args.QueryText.ToLower();

var results = ast.getList().Where(i => i.FIO.ToLower().Contains(term)).ToList();

List<String> destinationStrings = new List<string>();

foreach (Student f in results)

{

destinationStrings.Add(f.FIO);

}

suggestBox.ItemsSource = destinationStrings;

suggestBox.IsSuggestionListOpen = true;

}

private void suggestBox\_SuggestionChosen(AutoSuggestBox sender, AutoSuggestBoxSuggestionChosenEventArgs args)

{

var p = ast.getList().Where(g => g.FIO == Convert.ToString(args.SelectedItem));

ListStudent4.ItemsSource = p;

}

private void tg1\_Toggled(object sender, RoutedEventArgs e)

{

ToggleSwitch tgs = sender as ToggleSwitch;

if (tgs != null)

{

if (tgs.IsOn == true)

{

ListStudent2.ItemTemplate = (DataTemplate)this.Resources["editTemplate"];

}

else

{

ListStudent2.ItemTemplate = (DataTemplate)this.Resources["viewTemplate"];

}

}

}

private void FilterByMark\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

var p = ast.getList().Where(g => g.Mark1 == FilterByMark.SelectedValue.ToString() || g.Mark2 == FilterByMark.SelectedValue.ToString() || g.Mark3 == FilterByMark.SelectedValue.ToString());

ListStudent4.ItemsSource = p;

}

private void AddStudent\_Click(object sender, RoutedEventArgs e)

{

ast.addStudent(AddFioText.Text, AddMarkText1.SelectedValue.ToString(), AddMarkText2.SelectedValue.ToString(), AddMarkText3.SelectedValue.ToString());

ast.saveFile();

ListStudent1.ItemsSource = null;

ListStudent2.ItemsSource = null;

//ListStudent3.ItemsSource = null;

var tempList = ast.getList().OrderBy(p => p.FIO);//??

tempList.ToList().ForEach(q =>

{

ast.getList().Remove(q);

ast.getList().Add(q);

});

ListStudent1.ItemsSource = ast.getList();

ListStudent2.ItemsSource = ast.getList();

//ListStudent3.ItemsSource = ast.getList();

AddFioText.Text = "";

AddMarkText1.SelectedIndex = -1;

AddMarkText2.SelectedIndex = -1;

AddMarkText3.SelectedIndex = -1;

}

private void DeleteBtn\_Click(object sender, RoutedEventArgs e)

{

Button bt = sender as Button;

if (bt != null)

{

var p = bt.Parent as Grid;

var tt = (p.Children.Where(t => t.GetType().Name == "TextBlock").Last() as TextBlock).Text;

ast.deleteStudent(int.Parse(tt));

}

ast.saveFile();

}

private void LookPath\_Click(object sender, RoutedEventArgs e)

{

PathText.Text = ast.savePath();

}

}

}

**Код программы MainPage.xaml**

<forms:WindowsPage

x:Class="MobileLabs2.\_3.UWP.MainPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:forms="using:Xamarin.Forms.Platform.UWP"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:local="using:MobileLabs2.\_3.UWP"

Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

<!--

Паттерн MVVM

Binding

Bind

ObservableCollection

ListBox

Combobox & AutoSuggestBox

ListView

GridView

FlipView

Всё есть в примере, кроме FlipView & AutoSuggestBox, их добавить самому.

-->

<Page.Resources>

<local:Student x:Key="fl1" Id="4" FIO="Точилова Елизавета Александровна" Mark1="5" Mark2="5" Mark3="3"/>

<DataTemplate x:Key="viewTemplate">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="50" />

<ColumnDefinition Width="50" />

<ColumnDefinition Width="50" />

<ColumnDefinition Width="150" />

<ColumnDefinition Width="50" />

</Grid.ColumnDefinitions>

<TextBlock Text="{Binding Mark1}" Grid.Column="0" />

<TextBlock Text="{Binding Mark2}" Grid.Column="1" />

<TextBlock Text="{Binding Mark3}" Grid.Column="2" />

<TextBlock Text="{Binding FIO}" Grid.Column="3" />

<TextBlock Text="{Binding Id}" Grid.Column="4" />

</Grid>

</DataTemplate>

<DataTemplate x:Key="editTemplate">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="50" />

<ColumnDefinition Width="50" />

<ColumnDefinition Width="50" />

<ColumnDefinition Width="150" />

<ColumnDefinition Width="50" />

<ColumnDefinition Width="100" />

</Grid.ColumnDefinitions>

<TextBlock Text="{Binding Mark1}" Grid.Column="0" />

<TextBlock Text="{Binding Mark2}" Grid.Column="1" />

<TextBlock Text="{Binding Mark3}" Grid.Column="2" />

<TextBlock Text="{Binding FIO}" Grid.Column="3" />

<TextBlock Text="{Binding Id}" Grid.Column="4" />

<Button x:Name="DeleteBtn" Content="Удалить" Grid.Column="5" Click="DeleteBtn\_Click"/>

</Grid>

</DataTemplate>

</Page.Resources>

<Grid Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

<Grid>

<Grid.RowDefinitions>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

</Grid.ColumnDefinitions>

<StackPanel Grid.Row="0" Grid.Column="0">

<TextBlock Text="Вывод списка с использованием ListBox" Margin="10" FontSize="16" FontWeight="Bold"></TextBlock>

<ListBox x:Name="ListStudent1" ItemsSource="{x:Bind ast.getList()}" Margin="10">

<ListBox.ItemTemplate>

<DataTemplate x:DataType="local:Student">

<StackPanel Margin="5" Orientation="Horizontal">

<TextBlock FontSize="14" Width="40" Text="{Binding Id}" />

<TextBlock FontSize="14" Width="250" Text="{Binding FIO}"/>

<TextBlock FontSize="14" Width="40" Text="{Binding Mark1}"/>

<TextBlock FontSize="14" Width="40" Text="{Binding Mark2}"/>

<TextBlock FontSize="14" Width="40" Text="{Binding Mark3}"/>

</StackPanel>

</DataTemplate>

</ListBox.ItemTemplate>

</ListBox>

</StackPanel>

<StackPanel Grid.Row="0" Grid.Column="1">

<TextBlock Text="Вывод списка с использованием ListView" Margin="10" FontSize="16" FontWeight="Bold"></TextBlock>

<ToggleSwitch x:Name="tg1" Margin="5" Header="Режим редактирования" Toggled="tg1\_Toggled"></ToggleSwitch>

<ListView Name="ListStudent2" ItemsSource="{x:Bind ast.getList()}" Margin="10" ItemTemplate="{StaticResource viewTemplate}">

</ListView>

</StackPanel>

<StackPanel Grid.Row="1" Grid.Column="0">

<TextBlock Text="Фильтрация по оценкам" Margin="10" FontSize="16" FontWeight="Bold"></TextBlock>

<ComboBox x:Name="FilterByMark" Margin="10" SelectionChanged="FilterByMark\_SelectionChanged">

<x:String>1</x:String>

<x:String>2</x:String>

<x:String>3</x:String>

<x:String>4</x:String>

<x:String>5</x:String>

</ComboBox>

<AutoSuggestBox HorizontalAlignment="Left" Margin="10" x:Name="suggestBox" Header="Фильтрация по фамилии студента" PlaceholderText="Студенты" Width="242" QueryIcon="Find"

QuerySubmitted="suggestBox\_QuerySubmitted" SuggestionChosen="suggestBox\_SuggestionChosen"/>

<ListView Name="ListStudent4" Margin="10">

<ListView.ItemTemplate>

<DataTemplate>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="Auto" />

<ColumnDefinition Width="230" />

<ColumnDefinition Width="Auto" />

<ColumnDefinition Width="Auto" />

<ColumnDefinition Width="Auto" />

</Grid.ColumnDefinitions>

<TextBlock Text="{Binding Id}" Grid.Column="0" Margin="10 0" />

<TextBlock Text="{Binding FIO}" Grid.Column="1" Margin="10 0"/>

<TextBlock Text="{Binding Mark1}" Grid.Column="2" Margin="10 0"/>

<TextBlock Text="{Binding Mark2}" Grid.Column="3" Margin="10 0"/>

<TextBlock Text="{Binding Mark3}" Grid.Column="4" Margin="10 0"/>

</Grid>

</DataTemplate>

</ListView.ItemTemplate>

</ListView>

</StackPanel>

<StackPanel Grid.Row="1" Grid.Column="1">

<TextBlock Text="Добавление новой записи" Margin="10" FontSize="16" FontWeight="Bold"></TextBlock>

<TextBlock Text="Введите ФИО" Margin="10" FontSize="14"></TextBlock>

<TextBox x:Name="AddFioText" Margin="10 0 10 0"></TextBox>

<StackPanel Orientation="Horizontal">

<ComboBox x:Name="AddMarkText1" Margin="10 0 10 0" Header="Оценка по мат.">

<x:String>1</x:String>

<x:String>2</x:String>

<x:String>3</x:String>

<x:String>4</x:String>

<x:String>5</x:String>

</ComboBox>

<ComboBox x:Name="AddMarkText2" Margin="10 0 10 0" Header="Оценка по русс.">

<x:String>1</x:String>

<x:String>2</x:String>

<x:String>3</x:String>

<x:String>4</x:String>

<x:String>5</x:String>

</ComboBox>

<ComboBox x:Name="AddMarkText3" Margin="10 0 10 0" Header="Оценка по общ">

<x:String>1</x:String>

<x:String>2</x:String>

<x:String>3</x:String>

<x:String>4</x:String>

<x:String>5</x:String>

</ComboBox>

</StackPanel>

<RelativePanel>

<Button x:Name="AddStudent" Margin="10" Content="Добавить" Width="100" Click="AddStudent\_Click"></Button>

<Button x:Name="LookPath" Margin="10" RelativePanel.RightOf="AddStudent" Content="Смотреть путь до базы" Width="200" Click="LookPath\_Click"></Button>

<TextBlock x:Name ="PathText" TextWrapping="Wrap" Margin="10" RelativePanel.Below="AddStudent" FontSize="16" FontWeight="Bold"></TextBlock>

</RelativePanel>

</StackPanel>

</Grid>

</Grid>

</forms:WindowsPage>

**Код программы AllStudents.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Runtime.Serialization;

using System.Runtime.Serialization.Json;

using System.IO;

using Windows.Storage;

using System.Collections.ObjectModel;

namespace MobileLabs2.\_3.UWP

{

[DataContract]

internal class Student

{

[DataMember] public int Id { get; set; }

[DataMember] public string FIO { get; set; }

[DataMember] public string Mark1 { get; set; }

[DataMember] public string Mark2 { get; set; }

[DataMember] public string Mark3 { get; set; }

}

internal class allStudents

{

private ObservableCollection<Student> allStudentsLst;

private DataContractJsonSerializer jsonFormatter = new DataContractJsonSerializer(typeof(ObservableCollection<Student>));

public allStudents()

{

allStudentsLst = new ObservableCollection<Student>();

}

public void addStudent(string FIO, string mark1, string mark2, string mark3)

{

if (allStudentsLst.Count == 0)

{

allStudentsLst.Add(new Student { Id = 1, FIO = FIO, Mark1 = mark1, Mark2 = mark2, Mark3 = mark3});

}

else

{

allStudentsLst.Add(new Student { Id = int.Parse(allStudentsLst.Last().Id.ToString()) + 1, FIO = FIO, Mark1 = mark1, Mark2 = mark2, Mark3 = mark3 });

}

}

public ObservableCollection<Student> getList()

{

return allStudentsLst;

}

public string savePath()

{

StorageFolder localFolder = ApplicationData.Current.LocalFolder;

return string.Format("{0}\\{1}", localFolder.Path, "save.json");

}

public void saveFile()

{

var p = savePath();

using (FileStream fs = new FileStream(p, FileMode.Create))

{

jsonFormatter.WriteObject(fs, allStudentsLst);

}

}

public void openFile()

{

var p = savePath();

using (FileStream fs = new FileStream(p, FileMode.Open))

{

allStudentsLst = (ObservableCollection<Student>)jsonFormatter.ReadObject(fs);

}

}

public void deleteStudent(int id)

{

var delitem = allStudentsLst.Where(t => t.Id == id).First();

allStudentsLst.Remove(delitem);

}

//public void editStudent(int id, string fio)

//{

// var delitem = allStudentsLst.Where(t => t.Id == id).First();

// delitem.FIO = fio;

// allStudentsLst.saveFile(delitem);

//}

}

}