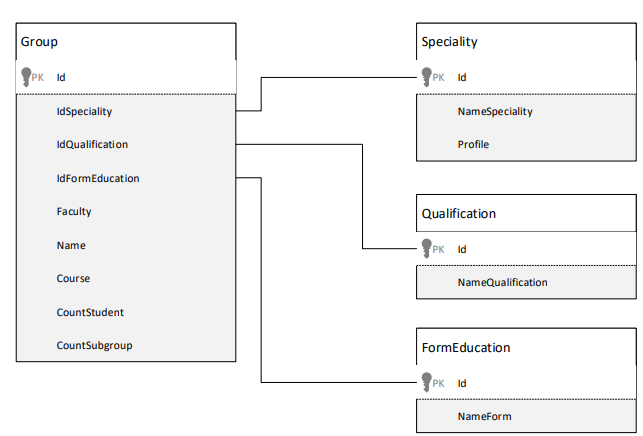
Вариант 11

Ссылка на GIT Репозиторий - https://github.com/ArtesianBeliever/WpfApp1



Назначение свойств класса Студенческая группа –Group:

• Id – идентификатор класса;

• IdSpeciality – свойство для связи с классом Speciality;

• IdQualification– свойство для связи с классом Qualification;

• IdFormEducation– свойство для связи с классом FormEducation;

• Faculty –факультет;

• Name – наименование группы;

• Course – курс;

• CountStudent – количество студентов.

• CountSubgrroup – количество подгрупп

Назначение свойств класса Направление подготовки –Speciality:

• ID – идентификатор класса;

• NameSpeciality – направление подготовки бакалавра;

• Profile – профиль подготовки.

Назначение свойств класса Квадификация –Qualification:

• ID – идентификатор класса;

• NameQualification – квалификация подготовки бакалавра;

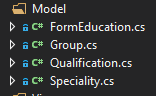
Назначение свойств класса Форма обучения – FormEducation:

• ID – идентификатор класса;

• NameForm– форма обучения.

1 Создание модели данных.

Создаем классы Group.cs, Speciality.cs, Qualification.cs, FormEducation.cs



Код Group.cs

class Group

{

public int Id { get; set; }

public int IdSpeciality { get; set; }

public int IdQualification { get; set; }

public int IdFormEducation { get; set; }

public string Faculty { get; set; }

public string Name { get; set; }

public int Course { get; set; }

public int CountStudent { get; set;}

public int CountSubgroup { get; set; }

public Group() { }

public Group(int id, int idSpeciality, int idQualification, int idFormEducation, string faculty,

string name, int course, int countStudent, int countSubgroup)

{

this.Id = id;

this.IdSpeciality = idSpeciality;

this.IdQualification = idQualification;

this.IdFormEducation = idFormEducation;

this.Faculty = faculty;

this.Name = name;

this.Course = course;

this.CountStudent = countStudent;

this.CountSubgroup = countSubgroup;

}

}

Код Speciality.cs

class Speciality

{

public int Id { get; set; }

public string NameSpeciality { get; set; }

public string Profile { get; set; }

public Speciality() { }

public Speciality(int id, string nameSpeciality, string profile)

{

this.Id = id;

this.NameSpeciality = nameSpeciality;

this.Profile = profile;

}

}

Код Qualification.cs

class Qualification

{

public int Id { get; set; }

public string NameQualification { get; set; }

public Qualification() { }

public Qualification (int id, string nameQualification)

{

this.Id = id;

this.NameQualification = nameQualification;

}

}

Код FormEducation.cs

class FormEducation

{

public int Id { get; set; }

public string NameForm { get; set; }

public FormEducation ( ) { }

public FormEducation ( int id, string nameForm)

{

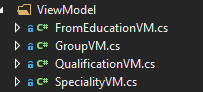
this.Id = id;

this.NameForm = nameForm;

}

}

Создаём классы для коллекций данных соответствующих классов.



class FormEducationVM

{

public ObservableCollection<FormEducation> ListQualification

{

get;

set;

} =

new ObservableCollection<FormEducation>();

public FormEducationVM()

{

this.ListQualification.Add(new FormEducation

{

Id = 1,

NameForm = "Очная"

});

this.ListQualification.Add(new FormEducation

{

Id = 2,

NameForm = "Заочная"

});

}

}

class GroupVM

{

public ObservableCollection<Group> ListGroup

{

get;

set;

} =

new ObservableCollection<Group>();

public GroupVM()

{

this.ListGroup.Add(

new Group

{

Id = 1,

IdSpeciality = 1,

IdQualification = 1,

IdFormEducation = 1,

Faculty = "KTiIB",

Name = "PIZS-321",

Course = 2,

CountStudent = 21,

CountSubgroup = 1

});

this.ListGroup.Add(

new Group

{

Id = 2,

IdSpeciality = 2,

IdQualification = 3,

IdFormEducation = 2,

Faculty = "KTiIB",

Name = "PRIZS-321",

Course = 2,

CountStudent = 24,

CountSubgroup = 2

});

this.ListGroup.Add(

new Group

{

Id = 3,

IdSpeciality = 3,

IdQualification = 2,

IdFormEducation = 2,

Faculty = "KTiIB",

Name = "PIZ-321",

Course = 2,

CountStudent = 11,

CountSubgroup = 2

});

}

}

class QualificationVM

{

public ObservableCollection<Qualification> ListQualification

{

get;

set;

} =

new ObservableCollection<Qualification>();

public QualificationVM()

{

this.ListQualification.Add(new Qualification

{

Id = 1,

NameQualification = "Специалист"

});

this.ListQualification.Add(new Qualification

{

Id = 2,

NameQualification = "Академический бакалавр"

});

this.ListQualification.Add(new Qualification

{

Id = 1,

NameQualification = "Прикладной бакалавр"

});

}

}

class SpecialityVM

{

public ObservableCollection<Speciality> ListSpeciality

{

get;

set;

} =

new ObservableCollection<Speciality>();

public SpecialityVM()

{

this.ListSpeciality.Add(new Speciality

{

Id = 1,

NameSpeciality = "Прикладная Информатика",

Profile = "Программист"

});

this.ListSpeciality.Add(new Speciality

{

Id = 2,

NameSpeciality = "Программная инженерия",

Profile = "Инженер"

});

this.ListSpeciality.Add(new Speciality

{

Id = 3,

NameSpeciality = "Компьютерные сети",

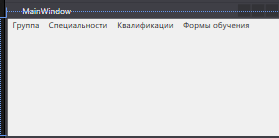
Profile = "Инженер сетей"

});

}

}

На главном окне приложения создадим меню для вывода и обработки данных.



<Window x:Class="WpfApp1.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:local="clr-namespace:WpfApp1"

mc:Ignorable="d"

Title="MainWindow" Height="200" Width="400">

<Grid>

<Menu>

<MenuItem x:Name="Group" Header="Группа" Click="Group\_OnClick"></MenuItem>

<MenuItem x:Name="Spec" Header="Специальности" Click="Spec\_OnClick"></MenuItem>

<MenuItem x:Name="Qual" Header="Квалификации" Click="Qual\_OnClick"></MenuItem>

<MenuItem x:Name="Form" Header="Формы обучения" Click="Form\_OnClick"></MenuItem>

</Menu>

</Grid>

</Window>

С# Код

namespace WpfApp1

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void Group\_OnClick(object sender, RoutedEventArgs e)

{

WindowGroup wGroup = new WindowGroup();

wGroup.Show();

}

private void Form\_OnClick(object sender, RoutedEventArgs e)

{

WindowForm wForm = new WindowForm();

wForm.Show();

}

private void Spec\_OnClick(object sender, RoutedEventArgs e)

{

WindowSpec wSpec = new WindowSpec();

wSpec.Show();

}

private void Qual\_OnClick(object sender, RoutedEventArgs e)

{

WindowQual wQual = new WindowQual();

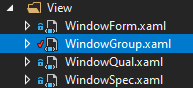
wQual.Show();

}

}

}

Создаём окна для предоставления данных



Код WindowGroup

<Window x:Class="WpfApp1.View.WindowGroup"

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:WpfApp1.View"

mc:Ignorable="d"

Title="Группы" Height="450" Width="800">

<Grid>

<StackPanel>

<ListView x:Name="lvGroup">

<ListView.View>

<GridView>

<GridView.Columns>

<GridViewColumn Header="Код" Width="50"

DisplayMemberBinding="{Binding Id}"/>

<GridViewColumn Header="Специальность"

DisplayMemberBinding="{Binding IdSpeciality}"/>

<GridViewColumn Header="Квалификация"

DisplayMemberBinding="{Binding IdQualification}"/>

<GridViewColumn Header="Форма обучения"

DisplayMemberBinding="{Binding IdFormEducation}"/>

<GridViewColumn Header="Факультет"

DisplayMemberBinding="{Binding Faculty}"/>

<GridViewColumn Header="Имя группы"

DisplayMemberBinding="{Binding Name}"/>

<GridViewColumn Header="Курс"

DisplayMemberBinding="{Binding Course}"/>

<GridViewColumn Header="Кол-во Студентов"

DisplayMemberBinding="{Binding CountStudent}"/>

<GridViewColumn Header="Кол-во Подгрупп"

DisplayMemberBinding="{Binding CountSubgroup}"/>

</GridView.Columns>

</GridView>

</ListView.View>

</ListView>

</StackPanel>

</Grid>

</Window>

C# код

public partial class WindowGroup : Window

{

public WindowGroup()

{

InitializeComponent();

GroupVM vmGroup = new GroupVM();

lvGroup.ItemsSource = vmGroup.ListGroup;

}

}

Делаем также с остальными окнами

<Window x:Class="WpfApp1.View.WindowSpec"

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:WpfApp1.View"

mc:Ignorable="d"

Title="Специальности" Height="450" Width="400">

<StackPanel Background="#FFD8D8D8">

<Label Margin="5" HorizontalAlignment="Center">Список Спецаильностей</Label>

<ListView x:Name="lvSpeciality" Background="#FFD3CDCD">

<ListView.View >

<GridView>

<GridView.Columns>

<GridViewColumn Header="Код" Width="50"

DisplayMemberBinding="{Binding Id}"/>

<GridViewColumn Header="Наименование Специальности"

DisplayMemberBinding="{Binding NameSpeciality}"/>

<GridViewColumn Header="Профиль"

DisplayMemberBinding="{Binding Profile}"/>

</GridView.Columns>

</GridView>

</ListView.View>

</ListView>

</StackPanel>

</Window>

public partial class WindowSpec : Window

{

public WindowSpec()

{

InitializeComponent();

SpecialityVM vmSpec = new SpecialityVM();

lvSpeciality.ItemsSource = vmSpec.ListSpeciality;

}

}

<Window x:Class="WpfApp1.View.WindowQual"

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:WpfApp1.View"

mc:Ignorable="d"

Title="WindowQual" Height="450" Width="400">

<StackPanel Background="#FFD8D8D8">

<Label Margin="5" HorizontalAlignment="Center">Список Спецаильностей</Label>

<ListView x:Name="lvQual" Background="#FFD3CDCD">

<ListView.View >

<GridView>

<GridView.Columns>

<GridViewColumn Header="Код" Width="50"

DisplayMemberBinding="{Binding Id}"/>

<GridViewColumn Header="Наименование Квалификации"

DisplayMemberBinding="{Binding NameQualification}"/>

</GridView.Columns>

</GridView>

</ListView.View>

</ListView>

</StackPanel>

</Window>

public partial class WindowQual : Window

{

public WindowQual()

{

InitializeComponent();

QualificationVM vmQual = new QualificationVM();

lvQual.ItemsSource = vmQual.ListQualification;

}

}

<Window x:Class="WpfApp1.View.WindowForm"

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:WpfApp1.View"

mc:Ignorable="d"

Title="Формы" Height="250" Width="300">

<StackPanel Background="#FFD8D8D8">

<Label Margin="5" HorizontalAlignment="Center">Список Форм</Label>

<ListView x:Name="lvForm" Background="#FFD3CDCD">

<ListView.View >

<GridView>

<GridView.Columns>

<GridViewColumn Header="Код" Width="50"

DisplayMemberBinding="{Binding Id}"/>

<GridViewColumn Header="Наименование Формы обучения"

DisplayMemberBinding="{Binding NameForm}"/>

</GridView.Columns>

</GridView>

</ListView.View>

</ListView>

</StackPanel>

</Window>

public partial class WindowForm : Window

{

public WindowForm()

{

InitializeComponent();

FormEducationVM vmForm = new FormEducationVM();

lvForm.ItemsSource = vmForm.ListFormEducation;

}

}

Далее необходимо сделать отображение Специальности, Квалификации и формы обучения вместо соотвествующего ID, для этого создаём классы FindSpec ,FindQual, FindForm.

class FindSpec

{

int id;

public FindSpec(int id)

{

this.id = id;

}

public bool SpecPredicate(Speciality speciality)

{

return speciality.Id == id;

}

}

class FindQual

{

int id;

public FindQual(int id)

{

this.id = id;

}

public bool QualPredicate(Qualification qualification)

{

return qualification.Id == id;

}

}

class FindForm

{

int id;

public FindForm(int id)

{

this.id = id;

}

public bool FormPredicate(FormEducation formEducation)

{

return formEducation.Id == id;

}

}

Затем создаём класс GroupDPO, который отличается от обычного класса тем, что вместо ID специальности, квалификации и формы обучения используется строка

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace WpfApp1.Model

{

class GroupDPO

{

public int Id { get; set; }

public string Speciality { get; set; }

public string Qualification { get; set; }

public string FormEducation { get; set; }

public string Faculty { get; set; }

public string Name { get; set; }

public int Course { get; set; }

public int CountStudent { get; set; }

public int CountSubgroup { get; set; }

public GroupDPO() { }

public GroupDPO(int id, string Speciality, string Qualification, string FormEducation, string faculty,

string name, int course, int countStudent, int countSubgroup)

{

this.Id = id;

this.Speciality = Speciality;

this.Qualification = Qualification;

this.FormEducation = FormEducation;

this.Faculty = faculty;

this.Name = name;

this.Course = course;

this.CountStudent = countStudent;

this.CountSubgroup = countSubgroup;

}

}

}

Далее редактируем GroupWindow следующим образом

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.Shapes;

using WpfApp1.ViewModel;

using WpfApp1.Model;

using WpfApp1.Helper;

using System.Collections.ObjectModel;

namespace WpfApp1.View

{

/// <summary>

/// Логика взаимодействия для WindowGroup.xaml

/// </summary>

public partial class WindowGroup : Window

{

public WindowGroup()

{

InitializeComponent();

GroupVM vmGroup = new GroupVM();

/// lvGroup.ItemsSource = vmGroup.ListGroup;

SpecialityVM vmSpecs = new SpecialityVM();

QualificationVM vmQuals = new QualificationVM();

FormEducationVM vmForms = new FormEducationVM();

List<Speciality> specs = new List<Speciality>();

List<Qualification> quals = new List<Qualification>();

List<FormEducation> forms = new List<FormEducation>();

foreach (Speciality s in vmSpecs.ListSpeciality)

{

specs.Add(s);

}

foreach (Qualification q in vmQuals.ListQualification)

{

quals.Add(q);

}

foreach (FormEducation f in vmForms.ListFormEducation)

{

forms.Add(f);

}

ObservableCollection<GroupDPO> groups = new ObservableCollection<GroupDPO>();

FindSpec finder1;

FindQual finder2;

FindForm finder3;

foreach (var p in vmGroup.ListGroup)

{

finder1 = new FindSpec(p.IdSpeciality);

Speciality spe = specs.Find(new Predicate<Speciality>(finder1.SpecPredicate));

finder2 = new FindQual(p.IdQualification);

Qualification qua = quals.Find(new Predicate<Qualification>(finder2.QualPredicate));

finder3 = new FindForm(p.IdFormEducation);

FormEducation forma = forms.Find(new Predicate<FormEducation>(finder3.FormPredicate));

groups.Add(new GroupDPO

{

Id = p.Id,

Speciality = spe.NameSpeciality,

Qualification = qua.NameQualification,

FormEducation = forma.NameForm,

Faculty = p.Faculty,

Name = p.Name,

Course = p.Course,

CountStudent = p.CountStudent,

CountSubgroup = p.CountSubgroup

});

}

lvGroup.ItemsSource = groups;

}

}

}

А также изменим следующие строки в .xaml файле для соответствующей привязки.

<GridViewColumn Header="Специальность"

DisplayMemberBinding="{Binding Speciality}"/>

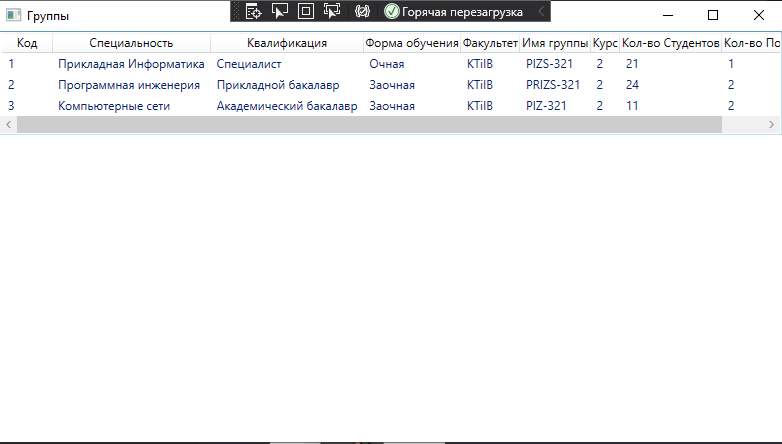
<GridViewColumn Header="Квалификация"

DisplayMemberBinding="{Binding Qualification}"/>

<GridViewColumn Header="Форма обучения"

DisplayMemberBinding="{Binding FormEducation}"/>

Результат:



Далее необходимо реализовать функцию добавления, редактирования и удаления записей. Для этого добавляем кнопки.

<StackPanel Orientation="Horizontal" Background="#FFE4E0E0">

<Button x:Name="btnAdd" Margin="10,10,5,10" Content="Добавить"

Height="25"

Click="btnAdd\_Click"/>

<Button x:Name="btnEdit" Margin="10,10,5,10" Content="Редактировать"

Height="25" Click="btnEdit\_Click"/>

<Button x:Name="btnDelete" Margin="10,10,5,10" Content="Удалить"

Height="25"

Click="btnDelete\_Click"/>

</StackPanel>

И сами обработчики кнопок для соответствующих окон.

public partial class WindowSpec : Window

{

SpecialityVM vmSpec = new SpecialityVM();

public WindowSpec()

{

InitializeComponent();

lvSpeciality.ItemsSource = vmSpec.ListSpeciality;

}

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

{

NewSpec wnSpec = new NewSpec

{

Title = "Новая должность",

Owner = this

};

int maxIdSpec = vmSpec.MaxIdS() + 1;

Speciality special = new Speciality

{

Id = maxIdSpec

};

wnSpec.DataContext = special;

if (wnSpec.ShowDialog() == true)

{

vmSpec.ListSpeciality.Add(special);

}

}

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

{

NewSpec wnRole = new NewSpec

{

Title = "Редактирование должности",

Owner = this

};

Speciality spec = lvSpeciality.SelectedItem as Speciality;

if (spec != null)

{

Speciality tempSpec = spec.ShallowCopy();

wnRole.DataContext = tempSpec;

if (wnRole.ShowDialog() == true)

{

// сохранение данных

spec.NameSpeciality = tempSpec.NameSpeciality;

spec.Profile = tempSpec.Profile;

lvSpeciality.ItemsSource = null;

lvSpeciality.ItemsSource = vmSpec.ListSpeciality;

}

}

else

{

MessageBox.Show("Необходимо выбрать должность для редактированния",

"Предупреждение", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

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

{

Speciality spec = (Speciality)lvSpeciality.SelectedItem;

if (spec != null)

{

MessageBoxResult result = MessageBox.Show("Удалить данные по должности: " +

spec.NameSpeciality, "Предупреждение", MessageBoxButton.OKCancel,

MessageBoxImage.Warning);

if (result == MessageBoxResult.OK)

{

vmSpec.ListSpeciality.Remove(spec);

}

}

else

{

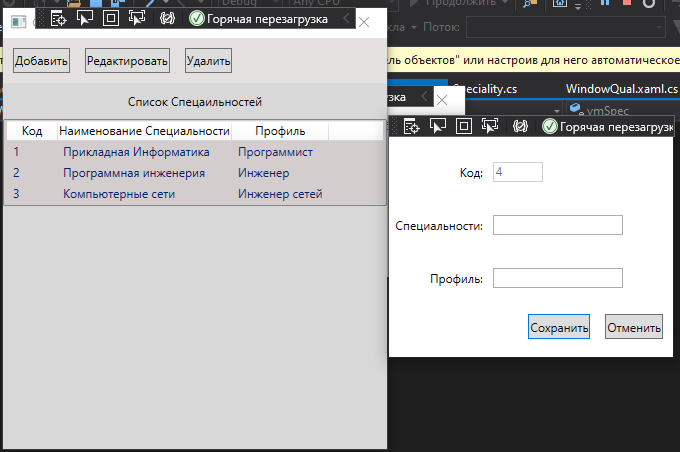
MessageBox.Show("Необходимо выбрать должность для удаления",

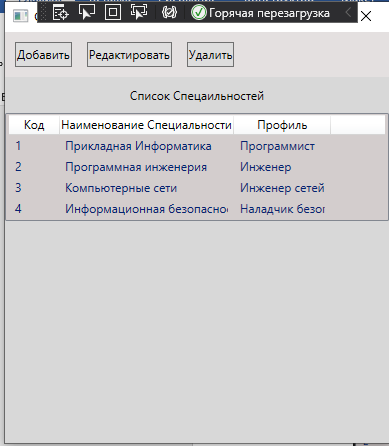
"Предупреждение", MessageBoxButton.OK, MessageBoxImage.Warning);

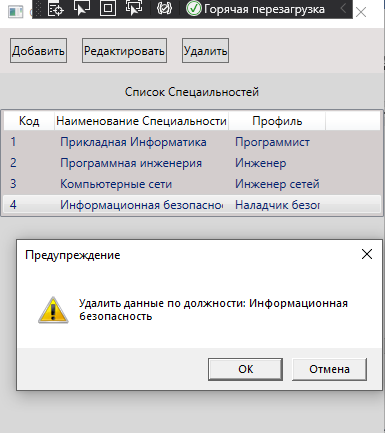
}

}

}







public partial class WindowQual : Window

{

QualificationVM vmQual = new QualificationVM();

public WindowQual()

{

InitializeComponent();

lvQual.ItemsSource = vmQual.ListQualification;

}

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

{

NewQual wnQual = new NewQual

{

Title = "Новая Квалификация",

Owner = this

};

// формирование кода новой должности

int maxIdQual = vmQual.MaxIdQ() + 1;

Qualification qual = new Qualification

{

Id = maxIdQual

};

wnQual.DataContext = qual;

if (wnQual.ShowDialog() == true)

{

vmQual.ListQualification.Add(qual);

}

}

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

{

NewQual wnQual = new NewQual

{

Title = "Редактирование Квалификации",

Owner = this

};

Qualification qual = lvQual.SelectedItem as Qualification;

if (qual != null)

{

Qualification tempQual = qual.ShallowCopy();

wnQual.DataContext = tempQual;

if (wnQual.ShowDialog() == true)

{

// сохранение данных

qual.NameQualification = tempQual.NameQualification;

lvQual.ItemsSource = null;

lvQual.ItemsSource = vmQual.ListQualification;

}

}

else

{

MessageBox.Show("Необходимо выбрать квалификацию для редактированния",

"Предупреждение", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

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

{

Qualification qual = (Qualification)lvQual.SelectedItem;

if (qual != null)

{

MessageBoxResult result = MessageBox.Show("Удалить данные по квалификации: " +

qual.NameQualification, "Предупреждение", MessageBoxButton.OKCancel,

MessageBoxImage.Warning);

if (result == MessageBoxResult.OK)

{

vmQual.ListQualification.Remove(qual);

}

}

else

{

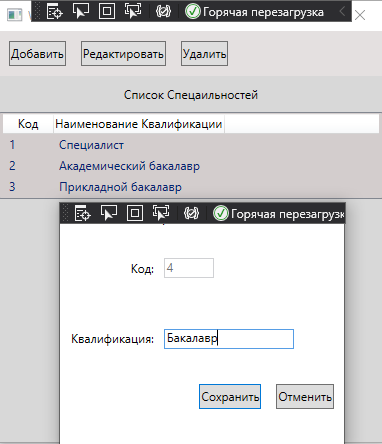
MessageBox.Show("Необходимо выбрать квалификацию для удаления",

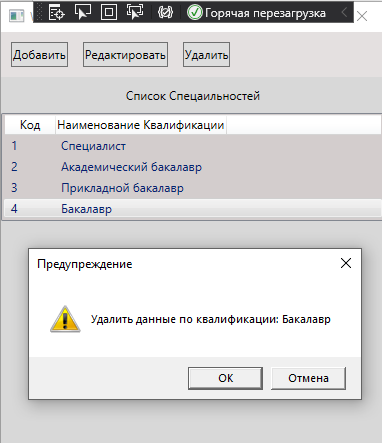
"Предупреждение", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

}





public partial class WindowForm : Window

{

FormEducationVM vmForm = new FormEducationVM();

public WindowForm()

{

InitializeComponent();

lvForm.ItemsSource = vmForm.ListFormEducation;

}

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

{

NewForm wnForm = new NewForm

{

Title = "Новая Форма",

Owner = this

};

// формирование кода новой формы

int maxIdForm = vmForm.MaxIdF() + 1;

FormEducation form = new FormEducation

{

Id = maxIdForm

};

wnForm.DataContext = form;

if (wnForm.ShowDialog() == true)

{

vmForm.ListFormEducation.Add(form);

}

}

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

{

NewForm wnForm = new NewForm

{

Title = "Редактирование Формы",

Owner = this

};

FormEducation form = lvForm.SelectedItem as FormEducation;

if (form != null)

{

FormEducation tempForm = form.ShallowCopy();

wnForm.DataContext = tempForm;

if (wnForm.ShowDialog() == true)

{

// сохранение данных

form.NameForm = tempForm.NameForm;

lvForm.ItemsSource = null;

lvForm.ItemsSource = vmForm.ListFormEducation;

}

}

else

{

MessageBox.Show("Необходимо выбрать Форму для редактированния",

"Предупреждение", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

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

{

FormEducation form = (FormEducation)lvForm.SelectedItem;

if (form != null)

{

MessageBoxResult result = MessageBox.Show("Удалить данные по форме: " +

form.NameForm, "Предупреждение", MessageBoxButton.OKCancel,

MessageBoxImage.Warning);

if (result == MessageBoxResult.OK)

{

vmForm.ListFormEducation.Remove(form);

}

}

else

{

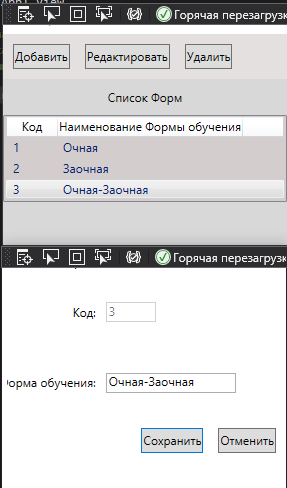
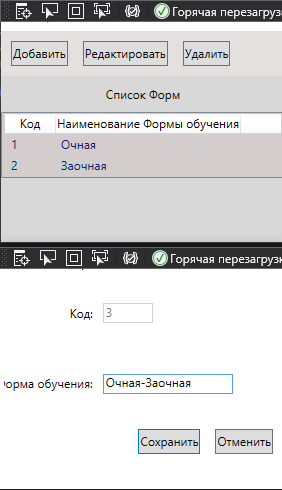
MessageBox.Show("Необходимо выбрать форму для удаления",

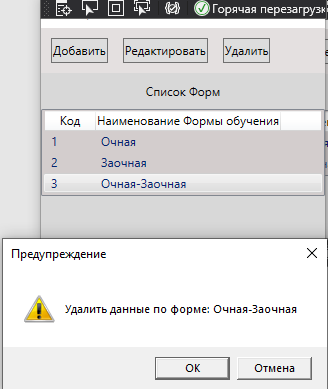
"Предупреждение", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

}





Далее добавим кнопки в окно групп.

<Button x:Name="btnAdd" Margin="10,10,5,10" Content="Добавить"

Height="25"

Click="btnAdd\_Click"/>

<Button x:Name="btnEdit" Margin="10,10,5,10" Content="Редактировать"

Height="25" Click="btnEdit\_Click"/>

<Button x:Name="btnDelete" Margin="10,10,5,10" Content="Удалить"

Height="25"

Click="btnDelete\_Click"/>

Для обработки этих кнопок понадобится дополнительные классы в классах Group и GroupDPO

public Group CopyFromPersonDPO(GroupDPO p)

{

SpecialityVM vmSpec = new SpecialityVM();

QualificationVM vmQual = new QualificationVM();

FormEducationVM vmForm = new FormEducationVM();

int specId = 0;

int qualId = 0;

int formId = 0;

foreach (var s in vmSpec.ListSpeciality)

{

if (s.NameSpeciality == p.Speciality)

{

specId = s.Id;

break;

}

}

foreach (var q in vmQual.ListQualification)

{

if (q.NameQualification == p.Qualification)

{

qualId = q.Id;

break;

}

}

foreach (var f in vmForm.ListFormEducation)

{

if (f.NameForm == p.FormEducation)

{

formId = f.Id;

break;

}

}

if (specId != 0 || formId != 0 || qualId!=0)

{

this.Id = p.Id;

this.IdSpeciality = IdSpeciality;

this.IdQualification = IdQualification;

this.IdFormEducation = IdFormEducation;

this.Faculty = p.Faculty;

this.Name = p.Name;

this.Course = p.Course;

this.CountStudent = p.CountStudent;

this.CountSubgroup = p.CountSubgroup;

}

return this;

}

public GroupDPO CopyFromGroup(Group group)

{

GroupDPO grDPO = new GroupDPO();

SpecialityVM vmSpec = new SpecialityVM();

QualificationVM vmQual = new QualificationVM();

FormEducationVM vmForm = new FormEducationVM();

string spec = string.Empty;

string qual = string.Empty;

string form = string.Empty;

foreach (var r in vmSpec.ListSpeciality)

{

if (r.Id == group.IdSpeciality)

{

spec = r.NameSpeciality;

break;

}

}

foreach (var q in vmQual.ListQualification)

{

if (q.Id == group.IdQualification)

{

qual = q.NameQualification;

break;

}

}

foreach (var f in vmForm.ListFormEducation)

{

if (f.Id == group.IdFormEducation)

{

form = f.NameForm;

break;

}

}

if (spec != string.Empty || qual != string.Empty || form != string.Empty)

{

grDPO.Id = group.Id;

grDPO.Speciality = spec;

grDPO.Qualification = qual;

grDPO.FormEducation = form;

grDPO.Faculty = group.Faculty;

grDPO.Name = group.Name;

grDPO.Course = group.Course;

grDPO.CountStudent = group.CountStudent;

grDPO.CountSubgroup = group.CountSubgroup;

}

return grDPO;

}

Далее создадим окно для добавления и редактирования.

<Window x:Class="WpfApp1.View.NewGroup"

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:WpfApp1.View"

mc:Ignorable="d"

Title="NewGroup" Height="800" Width="400">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="40\*"/>

<RowDefinition Height="40\*"/>

<RowDefinition Height="40\*"/>

<RowDefinition Height="40\*"/>

<RowDefinition Height="40\*"/>

<RowDefinition Height="40\*"/>

<RowDefinition Height="40\*"/>

<RowDefinition Height="40\*"/>

<RowDefinition Height="40\*"/>

<RowDefinition Height="50\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="40\*"/>

<ColumnDefinition Width="75\*"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Row="0" Grid.Column="0"

Text="Код:" HorizontalAlignment="Right"

VerticalAlignment="Center" Margin="5"/>

<TextBlock Grid.Row="1" Grid.Column="0"

Text="Специализация:" HorizontalAlignment="Right"

VerticalAlignment="Center" Margin="5"/>

<TextBlock Grid.Row="2" Grid.Column="0"

Text="Квалификация:" HorizontalAlignment="Right"

VerticalAlignment="Center" Margin="5"/>

<TextBlock Grid.Row="3" Grid.Column="0"

Text="Форма обучения:" HorizontalAlignment="Right"

VerticalAlignment="Center" Margin="5"/>

<TextBlock Grid.Row="4" Grid.Column="0"

Text="Факультет:" HorizontalAlignment="Right"

VerticalAlignment="Center" Margin="5"/>

<TextBlock Grid.Row="5" Grid.Column="0"

Text="Имя группы:" HorizontalAlignment="Right"

VerticalAlignment="Center" Margin="5"/>

<TextBlock Grid.Row="6" Grid.Column="0"

Text="Курс:" HorizontalAlignment="Right"

VerticalAlignment="Center" Margin="5"/>

<TextBlock Grid.Row="7" Grid.Column="0"

Text="Количество студентов:" HorizontalAlignment="Right"

VerticalAlignment="Center" Margin="5"/>

<TextBlock Grid.Row="8" Grid.Column="0"

Text="Количество Подгрупп:" HorizontalAlignment="Right"

VerticalAlignment="Center" Margin="5"/>

<TextBox x:Name="TbId" Grid.Row="0" Grid.Column="1"

Text="{Binding Id}" Height="20" Width="50"

HorizontalAlignment="Left" VerticalAlignment="Center"

Margin="5" IsEnabled="False"/>

<TextBox x:Name="TbFaculty" Grid.Row="4" Grid.Column="1"

Text="{Binding Faculty}"

Height="20" Width="130"

HorizontalAlignment="Left" VerticalAlignment="Center" Margin="5"/>

<TextBox x:Name="TbName" Grid.Row="5" Grid.Column="1"

Text="{Binding Name}"

Height="20" Width="130"

HorizontalAlignment="Left" VerticalAlignment="Center" Margin="5"/>

<TextBox x:Name="TbCourse" Grid.Row="6" Grid.Column="1"

Text="{Binding Course}"

Height="20" Width="130"

HorizontalAlignment="Left" VerticalAlignment="Center" Margin="5"/>

<TextBox x:Name="TbCountStudent" Grid.Row="7" Grid.Column="1"

Text="{Binding CountStudent}"

Height="20" Width="130"

HorizontalAlignment="Left" VerticalAlignment="Center" Margin="5"/>

<TextBox x:Name="TbCountSubgrroup" Grid.Row="8" Grid.Column="1"

Text="{Binding CountSubgroup}"

Height="20" Width="130"

HorizontalAlignment="Left" VerticalAlignment="Center" Margin="5"/>

<ComboBox x:Name="CbSpec" Grid.Row="1" Grid.Column="1" Height="20"

Width="130"

HorizontalAlignment="Left" VerticalAlignment="Center" Margin="5"

DisplayMemberPath="NameSpeciality"/>

<ComboBox x:Name="CbQual" Grid.Row="2" Grid.Column="1" Height="20"

Width="130"

HorizontalAlignment="Left" VerticalAlignment="Center" Margin="5"

DisplayMemberPath="NameQualification"/>

<ComboBox x:Name="CbForm" Grid.Row="3" Grid.Column="1" Height="20"

Width="130"

HorizontalAlignment="Left" VerticalAlignment="Center" Margin="5"

DisplayMemberPath="NameForm"/>

<StackPanel Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="10"

Orientation="Horizontal"

HorizontalAlignment="Right">

<Button x:Name="BtSave" Content="Сохранить" Height="25"

HorizontalAlignment="Right" VerticalAlignment="Top"

Margin="5,10,10,5"

Click="BtSave\_Click"/>

<Button x:Name="BtCansel" Content="Отменить" Height="25"

HorizontalAlignment="Right" VerticalAlignment="Top"

Margin="5,10,10,5" IsCancel="True"/>

</StackPanel>

</Grid>

</Window>

И наконец обработчики кнопок.

public partial class WindowGroup : Window

{

private GroupVM vmGroup;

private SpecialityVM vmSpec;

private QualificationVM vmQual;

private FormEducationVM vmForm;

private ObservableCollection<GroupDPO> grDPO;

private List<Speciality> specs;

private List<Qualification> quals;

private List<FormEducation> forms;

public WindowGroup()

{

InitializeComponent();

vmGroup = new GroupVM();

vmSpec = new SpecialityVM();

vmQual = new QualificationVM();

vmForm = new FormEducationVM();

specs = vmSpec.ListSpeciality.ToList();

quals = vmQual.ListQualification.ToList();

forms = vmForm.ListFormEducation.ToList();

// Формирование данных для отображения сотрудников с должностями

// на базе коллекции класса ListPerson<Person>

grDPO = new ObservableCollection<GroupDPO>();

foreach (var group in vmGroup.ListGroup)

{

GroupDPO p = new GroupDPO();

p = p.CopyFromGroup(group);

grDPO.Add(p);

}

lvGroup.ItemsSource = grDPO;

}

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

{

NewGroup wnEmployee = new NewGroup

{

Title = "Новая группа",

Owner = this

};

// формирование кода нового собрудника

int maxIdPerson = vmGroup.MaxId() + 1;

GroupDPO gr = new GroupDPO

{

Id = maxIdPerson,

};

wnEmployee.DataContext = gr;

wnEmployee.CbSpec.ItemsSource = specs;

wnEmployee.CbQual.ItemsSource = quals;

wnEmployee.CbForm.ItemsSource = forms;

if (wnEmployee.ShowDialog() == true)

{

Speciality s = (Speciality)wnEmployee.CbSpec.SelectedValue;

gr.Speciality = s.NameSpeciality;

Qualification q = (Qualification)wnEmployee.CbQual.SelectedValue;

gr.Qualification = q.NameQualification;

FormEducation f = (FormEducation)wnEmployee.CbForm.SelectedValue;

gr.FormEducation = f.NameForm;

grDPO.Add(gr);

Group p = new Group();

p = p.CopyFromPersonDPO(gr);

vmGroup.ListGroup.Add(p);

}

}

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

{

NewGroup wnEmployee = new NewGroup

{

Title = "Редактирование данных",

Owner = this

};

GroupDPO perDPO = (GroupDPO)lvGroup.SelectedValue;

GroupDPO tempPerDPO; // временный класс для редактирования

if (perDPO != null)

{

tempPerDPO = perDPO.ShallowCopy();

wnEmployee.DataContext = tempPerDPO;

wnEmployee.CbSpec.ItemsSource = specs;

wnEmployee.CbSpec.Text = tempPerDPO.Speciality;

wnEmployee.CbQual.ItemsSource = quals;

wnEmployee.CbQual.Text = tempPerDPO.Qualification;

wnEmployee.CbForm.ItemsSource = forms;

wnEmployee.CbForm.Text = tempPerDPO.FormEducation;

if (wnEmployee.ShowDialog() == true)

{

// перенос данных из временного класса в класс отображения данных

Speciality r = (Speciality)wnEmployee.CbSpec.SelectedValue;

perDPO.Speciality = r.NameSpeciality;

Qualification q = (Qualification)wnEmployee.CbQual.SelectedValue;

perDPO.Qualification = q.NameQualification;

FormEducation f = (FormEducation)wnEmployee.CbForm.SelectedValue;

perDPO.FormEducation = f.NameForm;

perDPO.Faculty = tempPerDPO.Faculty;

perDPO.Name = tempPerDPO.Name;

perDPO.Course = tempPerDPO.Course;

lvGroup.ItemsSource = null;

lvGroup.ItemsSource = grDPO;

// перенос данных из класса отображения данных в класс Person

FindGroup finder = new FindGroup(perDPO.Id);

List<Group> listPerson = vmGroup.ListGroup.ToList();

Group p = listPerson.Find(new Predicate<Group>(finder.PersonPredicate));

p = p.CopyFromPersonDPO(perDPO);

}

}

else

{

MessageBox.Show("Необходимо выбрать сотрудника для редактированния",

"Предупреждение", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

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

{

GroupDPO person = (GroupDPO)lvGroup.SelectedItem;

if (person != null)

{

MessageBoxResult result = MessageBox.Show("Удалить данные группы: \n" + person.Name + " " ,

"Предупреждение", MessageBoxButton.OKCancel, MessageBoxImage.Warning);

if (result == MessageBoxResult.OK)

{

// удаление данных в списке отображения данных

grDPO.Remove(person);

// удаление данных в списке классов ListPerson<Person>

Group per = new Group();

per = per.CopyFromPersonDPO(person);

vmGroup.ListGroup.Remove(per);

}

}

else

{

MessageBox.Show("Необходимо выбрать данные группы для удаления",

"Предупреждение", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

}

Результат

