**11** **Технологии создания графического пользовательского интерфейса**

Задание 1. Автоматизация учета сотрудников

Листинг задачи:

public partial class MainWindow : Window

{

private const string SavePath = "employees.json";

public ObservableCollection<Employee> Employees { get; set; } = new ObservableCollection<Employee>();

public ICommand AddEmployeeCommand { get; }

public ICommand EditEmployeeCommand { get; }

public ICommand DeleteEmployeeCommand { get; }

public MainWindow()

{

InitializeComponent();

DataContext = this;

LoadData();

AddEmployeeCommand = new RelayCommand(\_ => AddEmployee());

EditEmployeeCommand = new RelayCommand(\_ => EditEmployee(), \_ => dgEmployees.SelectedItem != null);

DeleteEmployeeCommand = new RelayCommand(\_ => DeleteEmployee(), \_ => dgEmployees.SelectedItem != null);

}

private void AddEmployee()

{

var dialog = new EmployeeWindow();

if (dialog.ShowDialog() == true)

{

Employees.Add(dialog.Employee);

SaveData();

}

}

private void EditEmployee()

{

var selected = dgEmployees.SelectedItem as Employee;

var dialog = new EmployeeWindow(selected);

if (dialog.ShowDialog() == true)

{

int index = Employees.IndexOf(selected);

Employees[index] = dialog.Employee;

SaveData();

}

}

private void DeleteEmployee()

{

var result = MessageBox.Show("Удалить сотрудника?", "Подтверждение",

MessageBoxButton.YesNo, MessageBoxImage.Warning);

if (result == MessageBoxResult.Yes && dgEmployees.SelectedItem is Employee employee)

{

Employees.Remove(employee);

SaveData();

}

}

private void LoadData()

{

if (File.Exists(SavePath))

{

var json = File.ReadAllText(SavePath);

Employees = JsonConvert.DeserializeObject<ObservableCollection<Employee>>(json)

?? new ObservableCollection<Employee>();

}

}

private void SaveData() =>

File.WriteAllText(SavePath, JsonConvert.SerializeObject(Employees, Formatting.Indented));

private void MenuItem\_Save\_Click(object sender, RoutedEventArgs e) => SaveData();

private void MenuItem\_Exit\_Click(object sender, RoutedEventArgs e) => Close();

public class Employee

{

public string FullName { get; set; }

public string Position { get; set; }

}

}

public class RelayCommand : ICommand

{

private readonly Action<object> \_execute;

private readonly Func<object, bool> \_canExecute;

public event EventHandler CanExecuteChanged

{

add => CommandManager.RequerySuggested += value;

remove => CommandManager.RequerySuggested -= value;

}

public RelayCommand(Action<object> execute, Func<object, bool> canExecute = null)

{

\_execute = execute;

\_canExecute = canExecute;

}

public bool CanExecute(object parameter) => \_canExecute?.Invoke(parameter) ?? true;

public void Execute(object parameter) => \_execute(parameter);

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| ФИО, Должность | Добавление сотрудника. Сообщение об ошибке |

Анализ результатов:

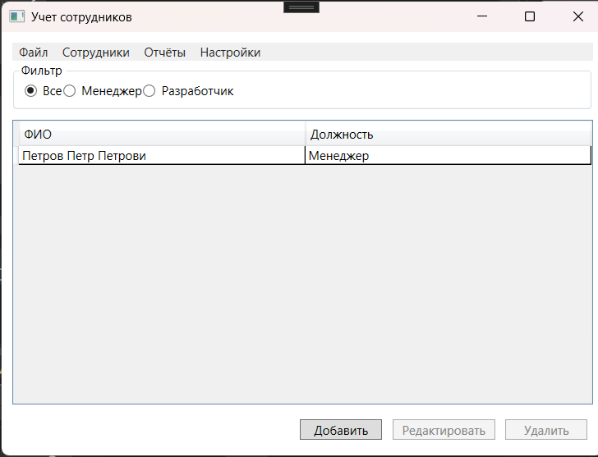


Рисунок 1.1 – Результат работы программы