­

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное

учреждения высшего образования

«Национальный исследовательский технологический университет «МИСиС»

Кафедра бизнес-информатики и систем управления производством

Курсовая работа

по дисциплине

«Объектно-ориентированное программирование»

на тему:

«Создание приложения Windows Forms»

Направление подготовки:

38.03.05 «Бизнес-информатика»

Семестр 4

|  |  |  |
| --- | --- | --- |
| Выполнил: | Проверила: |  |
| Касаткин А.Д  (Ф.И.О. студента) | Куликовский М.А  (Ф.И.О преподавателя) |  |
| ББИ-22-БА-3  (№ группы) | \_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оценка) |  |
| \_\_\_\_\_\_\_\_\_\_\_\_\_  (дата сдачи) | \_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата проверки) |  |
| Подпись: \_\_\_\_\_\_\_\_\_\_\_\_ | Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_ |  |

Москва – 2024­­

**Содержание**

[Цель работы 2](#_Toc185368590)

[Основная часть 2](#_Toc185368591)

[Заключение 6](#_Toc185368592)

[Использованные источники 7](#_Toc185368593)

Цель работы

Создать в Visual Studio решение из шаблона Windows Forms, которое сможет составлять связанные списки, выбирая заданные столбцы из базы данных. После чего показывать эти списки в виде таблиц и визуализировать данные через диаграмму.

Основная часть

**Техническое задание:**

* Проект должен содержать класс и массив экземпляров класса (или List).
* В работе необходимо выполнить построение графиков (диаграмм) и выполнить запросы к данным.
* Для выполнения запросов необходимо использовать язык Linq или методы расширения интерфейса IEnumerable.
* Источником данных для запросов является массив экземпляров класса или List.

**Описание проекта:**

Решение с новаторским названием NorthwindApp добавлена шаблонная база данных northwind.db и пакеты System.Data.SQLite и LiveChartsCore.SkiaSharpView.WinForm. При запуске приложения, пользователь видит форму, созданную с использованием шаблона Forms.

В этой форме:

* Таблица с данными, которая содержит часть данных всех работников из базы.
* Таблица с данными, которая содержит данные, соответствующие одному из двух условий на выбор.
* Выпадающий список с двумя условиями, выбирая между которыми пользователь обновляет вторую таблицу.
* График, показывающий экстеншен работников, которые подходят по условиям.

Выбираемые данные и условия заданы через код, пользователь не может на них повлиять. Работа также предполагает наличие у пользователя понимания, что такое экстеншен работника.

**Техническое описание:**

Ниже представлен код класса **Employees.cs**, в котором объявлены свойства полей и конструктор для полей.

namespace NorthwindApp

{

class Employees

{

private int \_employeeID; // Поле для свойства employeeID

private string \_lastName; // Поле для свойства lastName

private string \_firstName; // Поле для свойства firstName

private int \_extension; // Поле для свойства extension

private string \_country; // Поле для свойства country

public Employees(int employeeID, string lastName, string firstName, int extension, string country)

{

\_employeeID = employeeID;

\_lastName = lastName;

\_firstName = firstName;

\_extension = extension;

\_country = country;

}

// Свойства

public int employeeID { get => \_employeeID; set => \_employeeID = value; }

public string lastName { get => \_lastName; set => \_lastName = value; }

public string firstName { get => \_firstName; set => \_firstName = value; }

public int extension { get => \_extension; set => \_extension = value; }

public string country { get => \_country; set => \_country = value; }

}

}

Ниже представлен код класса Form1.cs, который вызывает класс Employees.cs, содержит методы для:

* Инициализации формы и её составляющих
* Считывания нажатия кнопки и загрузки данных в форму
* Обновления графиков при переключении выпадающего списка
* Считывания данных из таблицы и преобразование одного из значений

using System;

using System.Collections.Generic;

using System.Data.SQLite;

using System.Linq;

using System.Windows.Forms;

using LiveChartsCore.SkiaSharpView.WinForms;

using LiveChartsCore.SkiaSharpView;

using LiveChartsCore;

namespace NorthwindApp

{

public partial class Form1 : Form

{

private List<Employees> employees = new List<Employees>();

private CartesianChart chart;

public Form1()

{

InitializeComponent();

SetupChart();

}

private void SetupChart()

{

chart = new CartesianChart

{

Dock = DockStyle.Fill

};

panelChart.Controls.Add(chart);

}

private void Form1\_Load(object sender, EventArgs e)

{

ReloadProducts();

}

// Универсальный метод фильтрации

private void ApplyCurrentFilter()

{

if (comboBox1.SelectedIndex < 0)

{

System.Diagnostics.Debug.WriteLine("ComboBox не имеет выбранного элемента.");

return;

}

List<Employees> filteredEmployees = null;

// Выполняем фильтрацию

if (comboBox1.SelectedIndex == 0)

{

filteredEmployees = employees.Where(p => p.country == "USA").ToList();

}

else if (comboBox1.SelectedIndex == 1)

{

filteredEmployees = employees.Where(p => p.extension > 999).ToList();

}

// Обновляем DataGridView2

dataGridView2.DataSource = null;

dataGridView2.DataSource = filteredEmployees;

// Обновляем график

UpdateChart(filteredEmployees);

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

ApplyCurrentFilter();

}

private void UpdateChart(List<Employees> filteredEmployees)

{

if (filteredEmployees == null || filteredEmployees.Count == 0)

{

chart.Series = new ISeries[0];

return;

}

chart.Series = new ISeries[]

{

new ColumnSeries<int>

{

Values = filteredEmployees.Select(p => p.extension).ToArray(),

Name = "Extension (значение больше 999)"

}

};

chart.XAxes = new[]

{

new Axis

{

Labels = filteredEmployees.Select(p => p.lastName).ToArray()

}

};

System.Diagnostics.Debug.WriteLine("График обновлён.");

}

private void btnAddEmployees\_Click(object sender, EventArgs e)

{

ReloadProducts();

ApplyCurrentFilter();

}

private void ReloadProducts()

{

employees.Clear();

using (var connection = new SQLiteConnection("Data Source=northwind.db;Version=3;"))

{

connection.Open();

var command = new SQLiteCommand("SELECT EmployeeID, LastName, FirstName, Extension, Country FROM Employees", connection);

using (var reader = command.ExecuteReader())

{

while (reader.Read())

{

// Чтение данных через свойства

int employeeID = reader.GetInt32(0);

string lastName = reader.GetString(1);

string firstName = reader.GetString(2);

// Преобразование текста из Extension в int

int extension = 0;

if (int.TryParse(reader.GetString(3), out int parsedExtension))

{

extension = parsedExtension;

}

string country = reader.GetString(4);

// Добавление сотрудника в список

employees.Add(new Employees(employeeID, lastName, firstName, extension, country));

}

}

}

// Обновляем DataGridView1

dataGridView1.DataSource = null;

dataGridView1.DataSource = employees

.Select(e => new

{

ID = e.employeeID,

Фамилия = e.lastName,

Имя = e.firstName,

Extension = e.extension,

Страна = e.country

}).ToList();

// Обновляем ComboBox

comboBox1.Items.Clear();

comboBox1.Items.Add("Работники из USA");

comboBox1.Items.Add("Extension > 999");

// Устанавливаем первый элемент и применяем фильтрацию

comboBox1.SelectedIndex = 0;

ApplyCurrentFilter();

}

}

}

**Тестирование решения**

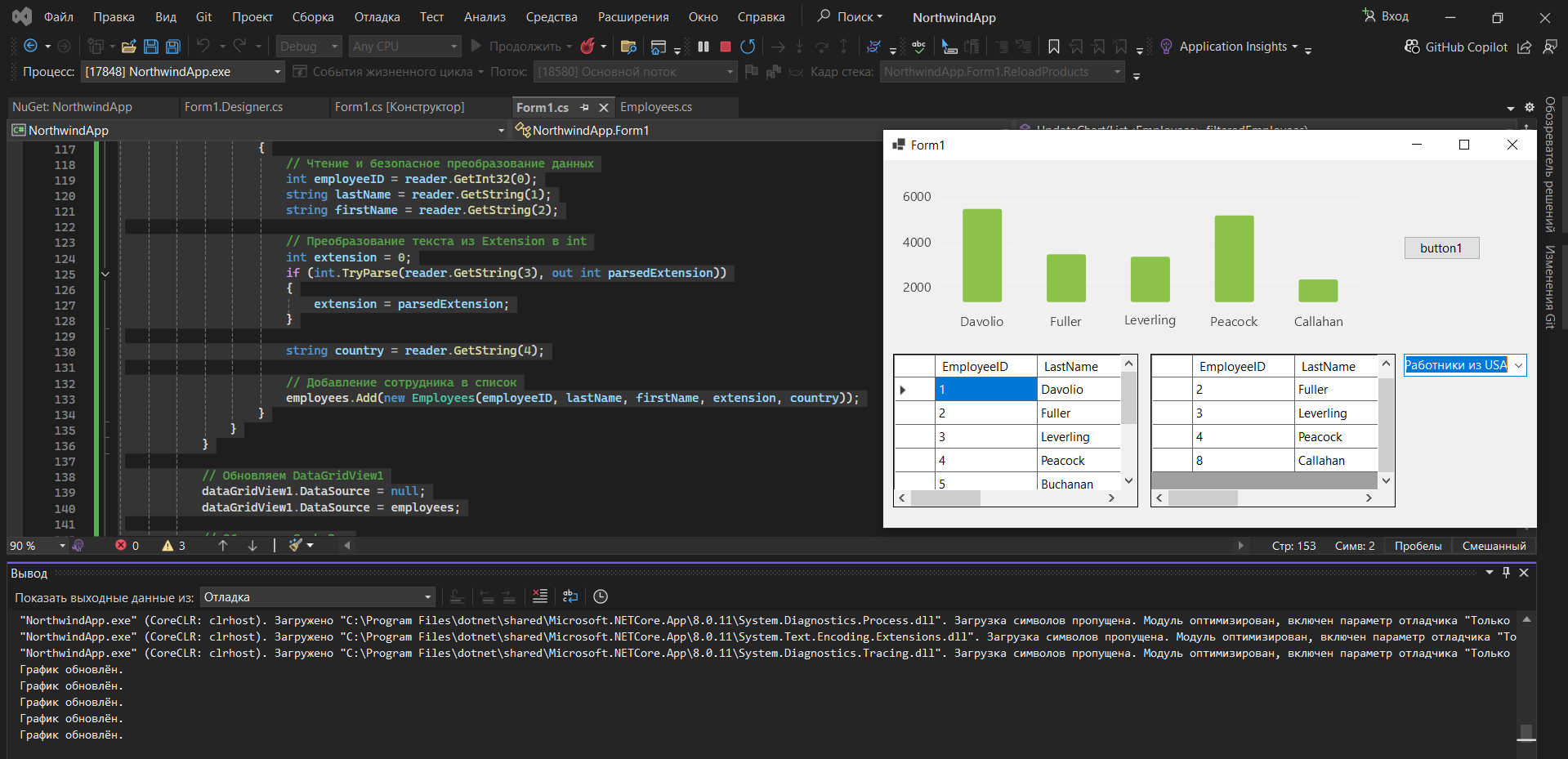


Рисунок 1 – Запущенная форма

Заключение

В ходе работы было создано решение из шаблона Windows Forms, которое может составлять связанные списки, выбирая заданные столбцы из базы данных. После чего показывать эти списки в виде таблиц и визуализировать данные через диаграмму.

Были получены навыки создания методов и массивов, использования Visual Studio и подключения к обозревателю кода пакетов.

Использованные источники

* ChatGpt