Министерство образования Республики Беларусь

Оршанский колледж ВГУ имени П.М. Машерова

Отчет

По модулю №6

« Графический интерфейс»

по учебной практике по программированию

Выполнил учащийся Войтеховская К.В.

группа 3ПОИС23 29.09.2025 г.

Проверил Алейников М.А.

30.09.2025 г.

Орша, 2025

**Ход работы.**

**Выполнение практических заданий.**

**Выполнение задания 1.** Разработайте систему учета задач с использованием базы данных. Пользователь должен иметь возможность добавлять, удалять и обновлять задачи.

Для выполнения задания было решено использовать БД в Access. Там в конструкторе таблиц были созданы поля представленные на рисунке 1.

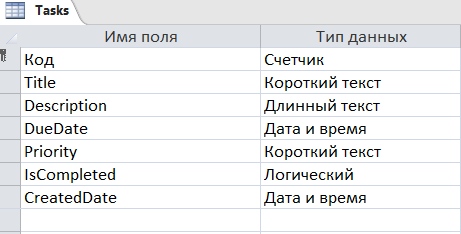
****

Рисунок 1 – Поля в таблице в БД

После этого БД была размещена в папке bin/debug созданного проекта и подключена через «Обозреватель решений». После этого на форме были размещены компоненты и написан код представленный на листинге 1.

Класс Form1 в программе содержит приватные поля для управления подключением к базе данных и отслеживания состояния приложения. Поле connectionString хранит параметры подключения к базе данных Access, в том числе путь к файлу базы данных. Поле connection представляет активное соединение с базой данных, а selectedTaskId хранит идентификатор выбранной в данный момент задачи для операций редактирования и удаления.

Конструктор класса выполняет инициализацию компонентов формы, вызывает метод InitializeDatabase для установки соединения с базой данных, загружает начальный набор задач через метод LoadTasks и настраивает интерфейс выбора приоритета с помощью метода SetupPriorityComboBox. Метод InitializeDatabase реализует подключение к источнику данных с обработкой исключительных ситуаций. Он создаёт экземпляр класса OleDbConnection, устанавливает физическое соединение с базой данных и перехватывает возможные ошибки подключения, информируя пользователя о возникших проблемах через модальные сообщения. Метод LoadTasks обеспечивает загрузку и отображение данных из таблицы Tasks. Он использует комбинацию классов OleDbDataAdapter и DataTable для извлечения и временного хранения данных. Особенностью реализации является динамическая настройка отображения колонок DataGridView, включая локализацию заголовков, настройку ширины столбцов и скрытие технических полей, что демонстрирует работу с метаданными результирующего набора.

Метод dataGridView1\_SelectionChanged обрабатывает событие изменения выбора в таблице задач и демонстрирует работу с элементами управления DataGridView. Он извлекает данные из выбранной строки, преобразует типы значений и заполняет соответствующие элементы формы, обеспечивая двустороннее взаимодействие между табличным представлением и формой редактирования. Метод ClearForm выполняет сброс состояния формы, включая очистку текстовых полей, установку значений по умолчанию для элементов управления и снятие выделения в таблице. Этот метод используется в нескольких операциях. Метод dataGridView1\_CellFormatting реализует визуальное оформление таблицы на основе данных. Он анализирует значение поля IsCompleted и изменяет цветовую схему строки, обеспечивая визуальную индикацию статуса выполнения задачи. Метод textBox1\_KeyPress демонстрирует обработку пользовательского ввода с клавиатуры и реализацию альтернативного способа взаимодействия через горячие клавиши. Он анализирует состояние приложения и динамически выбирает действие для выполнения в зависимости от контекста.

Листинг 1. Код программы приложения для рисования на холсте

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

namespace WindowsFormsApp1

{

public partial class Form1 : Form

{

private string connectionString = @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=|DataDirectory|\TasksDatabase.accdb";

private OleDbConnection connection;

private int selectedTaskId = -1;

public Form1()

{

InitializeComponent();

InitializeDatabase();

LoadTasks();

SetupPriorityComboBox();

}

private void InitializeDatabase()

{

try

{

connection = new OleDbConnection(connectionString);

connection.Open();

}

catch (Exception ex)

{

MessageBox.Show("Ошибка подключения к базе данных: " + ex.Message);

}

}

private void SetupPriorityComboBox()

{

comboBox1.Items.AddRange(new object[] { "Высокий", "Средний", "Низкий" });

comboBox1.SelectedIndex = 1; // Средний по умолчанию

}

private void LoadTasks()

{

try

{

string query = "SELECT \* FROM Tasks ORDER BY DueDate";

OleDbDataAdapter adapter = new OleDbDataAdapter(query, connection);

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

dataGridView1.DataSource = dataTable;

// Настройка внешнего вида DataGridView

if (dataGridView1.Columns.Count > 0)

{

// Скрытие колонки "Код"

if (dataGridView1.Columns.Contains("Код"))

{

dataGridView1.Columns["Код"].Visible = false;

}

if (dataGridView1.Columns.Contains("Title"))

{

dataGridView1.Columns["Title"].HeaderText = "Название";

dataGridView1.Columns["Title"].Width = 150;

}

if (dataGridView1.Columns.Contains("Description"))

{

dataGridView1.Columns["Description"].HeaderText = "Описание";

dataGridView1.Columns["Description"].Width = 200;

}

if (dataGridView1.Columns.Contains("DueDate"))

{

dataGridView1.Columns["DueDate"].HeaderText = "Срок выполнения";

dataGridView1.Columns["DueDate"].Width = 120;

}

if (dataGridView1.Columns.Contains("Priority"))

{

dataGridView1.Columns["Priority"].HeaderText = "Приоритет";

dataGridView1.Columns["Priority"].Width = 80;

}

if (dataGridView1.Columns.Contains("IsCompleted"))

{

dataGridView1.Columns["IsCompleted"].HeaderText = "Выполнена";

dataGridView1.Columns["IsCompleted"].Width = 80;

}

if (dataGridView1.Columns.Contains("CreatedDate"))

{

dataGridView1.Columns["CreatedDate"].HeaderText = "Дата создания";

dataGridView1.Columns["CreatedDate"].Width = 120;

}

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка загрузки задач: " + ex.Message);

}

}

private void button1\_Click(object sender, EventArgs e) // Добавить

{

if (string.IsNullOrWhiteSpace(textBox1.Text))

{

MessageBox.Show("Введите название задачи!");

return;

}

try

{

string query = @"INSERT INTO Tasks (Title, Description, DueDate, Priority, IsCompleted, CreatedDate)

VALUES (@Title, @Description, @DueDate, @Priority, @IsCompleted, @CreatedDate)";

using (OleDbCommand command = new OleDbCommand(query, connection))

{

// Явно указываем типы данных для параметров

command.Parameters.Add("@Title", OleDbType.VarWChar).Value = textBox1.Text.Trim();

command.Parameters.Add("@Description", OleDbType.VarWChar).Value = textBox2.Text ?? "";

command.Parameters.Add("@DueDate", OleDbType.Date).Value = dateTimePicker1.Value;

command.Parameters.Add("@Priority", OleDbType.VarWChar).Value = comboBox1.SelectedItem?.ToString() ?? "Средний";

command.Parameters.Add("@IsCompleted", OleDbType.Boolean).Value = checkBox1.Checked;

command.Parameters.Add("@CreatedDate", OleDbType.Date).Value = DateTime.Now;

int result = command.ExecuteNonQuery();

if (result > 0)

{

MessageBox.Show("Задача добавлена успешно!");

ClearForm();

LoadTasks();

}

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка добавления задачи: " + ex.Message);

}

}

private void button2\_Click(object sender, EventArgs e) // Обновить

{

if (selectedTaskId == -1)

{

MessageBox.Show("Выберите задачу для обновления!");

return;

}

if (string.IsNullOrWhiteSpace(textBox1.Text))

{

MessageBox.Show("Введите название задачи!");

return;

}

try

{

string query = @"UPDATE Tasks SET Title = @Title, Description = @Description, DueDate = @DueDate,

Priority = @Priority, IsCompleted = @IsCompleted WHERE Код = @Код";

using (OleDbCommand command = new OleDbCommand(query, connection))

{

command.Parameters.Add("@Title", OleDbType.VarWChar).Value = textBox1.Text.Trim();

command.Parameters.Add("@Description", OleDbType.VarWChar).Value = textBox2.Text ?? "";

command.Parameters.Add("@DueDate", OleDbType.Date).Value = dateTimePicker1.Value;

command.Parameters.Add("@Priority", OleDbType.VarWChar).Value = comboBox1.SelectedItem?.ToString() ?? "Средний";

command.Parameters.Add("@IsCompleted", OleDbType.Boolean).Value = checkBox1.Checked;

command.Parameters.Add("@Код", OleDbType.Integer).Value = selectedTaskId;

int result = command.ExecuteNonQuery();

if (result > 0)

{

MessageBox.Show("Задача обновлена успешно!");

ClearForm();

LoadTasks();

}

else

{

MessageBox.Show("Задача не найдена или не была обновлена.");

}

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка обновления задачи: " + ex.Message);

}

}

private void button3\_Click(object sender, EventArgs e) // Удалить

{

if (selectedTaskId == -1)

{

MessageBox.Show("Выберите задачу для удаления!");

return;

}

if (MessageBox.Show("Вы уверены, что хотите удалить эту задачу?", "Подтверждение удаления",

MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

try

{

string query = "DELETE FROM Tasks WHERE Код = @Код";

using (OleDbCommand command = new OleDbCommand(query, connection))

{

command.Parameters.Add("@Код", OleDbType.Integer).Value = selectedTaskId;

int result = command.ExecuteNonQuery();

if (result > 0)

{

MessageBox.Show("Задача удалена успешно!");

ClearForm();

LoadTasks();

}

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка удаления задачи: " + ex.Message);

}

}

}

private void button4\_Click(object sender, EventArgs e) // Обновить список

{

LoadTasks();

ClearForm();

}

private void button5\_Click(object sender, EventArgs e) // Очистить

{

ClearForm();

}

private void dataGridView1\_SelectionChanged(object sender, EventArgs e)

{

if (dataGridView1.SelectedRows.Count > 0 && dataGridView1.SelectedRows[0].Index >= 0)

{

DataGridViewRow selectedRow = dataGridView1.SelectedRows[0];

// Проверка, что строка не пустая и не является новой строкой

if (selectedRow.Cells["Код"].Value != null &&

selectedRow.Cells["Код"].Value != DBNull.Value)

{

selectedTaskId = Convert.ToInt32(selectedRow.Cells["Код"].Value);

// Заполнение формы данными выбранной задачи

textBox1.Text = selectedRow.Cells["Title"].Value?.ToString() ?? "";

textBox2.Text = selectedRow.Cells["Description"].Value?.ToString() ?? "";

if (selectedRow.Cells["DueDate"].Value != null &&

selectedRow.Cells["DueDate"].Value != DBNull.Value)

{

dateTimePicker1.Value = Convert.ToDateTime(selectedRow.Cells["DueDate"].Value);

}

string priority = selectedRow.Cells["Priority"].Value?.ToString() ?? "Средний";

comboBox1.SelectedItem = priority;

if (selectedRow.Cells["IsCompleted"].Value != null &&

selectedRow.Cells["IsCompleted"].Value != DBNull.Value)

{

checkBox1.Checked = Convert.ToBoolean(selectedRow.Cells["IsCompleted"].Value);

}

}

}

}

private void ClearForm()

{

textBox1.Clear();

textBox2.Clear();

dateTimePicker1.Value = DateTime.Now.AddDays(1);

comboBox1.SelectedIndex = 1; // Средний приоритет

checkBox1.Checked = false;

selectedTaskId = -1;

// Снятие выделения с DataGridView

dataGridView1.ClearSelection();

}

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

{

dateTimePicker1.Value = DateTime.Now.AddDays(1);

dataGridView1.ClearSelection();

}

private void Form1\_FormClosing(object sender, FormClosingEventArgs e)

{

connection?.Close();

connection?.Dispose();

}

private void textBox1\_KeyPress(object sender, KeyPressEventArgs e)

{

if (e.KeyChar == (char)Keys.Enter)

{

if (selectedTaskId == -1)

button1.PerformClick(); // Добавить

else

button2.PerformClick(); // Обновить

e.Handled = true;

}

}

private void dataGridView1\_CellFormatting(object sender, DataGridViewCellFormattingEventArgs e)

{

if (e.RowIndex < 0 || e.RowIndex >= dataGridView1.Rows.Count) return;

DataGridViewRow row = dataGridView1.Rows[e.RowIndex];

// Проверка существования колонки IsCompleted

if (dataGridView1.Columns.Contains("IsCompleted") &&

row.Cells["IsCompleted"].Value != null &&

row.Cells["IsCompleted"].Value != DBNull.Value)

{

try

{

bool isCompleted = Convert.ToBoolean(row.Cells["IsCompleted"].Value);

if (isCompleted)

{

row.DefaultCellStyle.BackColor = Color.LightGreen;

row.DefaultCellStyle.ForeColor = Color.DarkGreen;

}

else

{

row.DefaultCellStyle.BackColor = Color.White;

row.DefaultCellStyle.ForeColor = Color.Black;

}

}

catch

{

// Ошибки преобразования игнорируются

}

}

}

private void dateTimePicker1\_ValueChanged(object sender, EventArgs e)

{

}

}

}

На рисунке 2 представлен результат выполнения работы. Все кнопки выполняют своё функциональное назначение, задачи сохраняются и отображаются при повторных запусках программы.

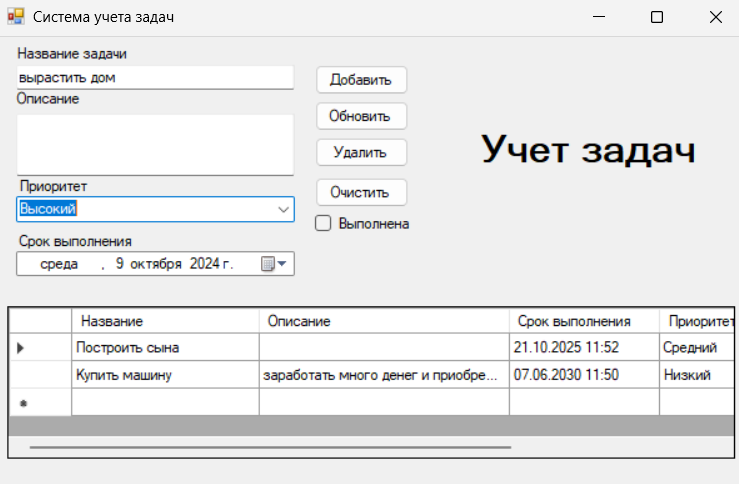


Рисунок 2 – Работа программы