МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «Гомельский государственный технический

университет имени П.О. Сухого»

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

Отчёт по лабораторной работе №7

По дисциплине «Объектно-ориентированное программирование»

**«****Технологии доступа к базам данных»**

Выполнил:

Студент группы ИТП-21

Коркуц С. И.

Принял:

ассистент Карабчикова Е.А.

Гомель 2020

**Цель работы:** изучить технологию доступа к базам данным.

**Задание:**

Реализовать CRUD операции используя технологию ADO.NET. В этой лабораторной работе запрещено использовать компоненты, которые скрывают работу с ADO.NET.

• Работу с БД реализовать через библиотеку классов, приложение должно содержать только вызовы соответствующих методов.

• Выделенный курсивом параметр расширить до справочной таблицы.

• БД содержать не менее 3х таблиц.

• Обязательно использовать подгрузку данных в элементы управления из справочных таблиц (Например: выпадающие списки).

• Провести модульное тестирование созданных библиотек.

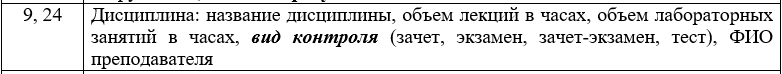


Рисунок 1 – Вариант задания

На рисунке 2 изображена структура решения.

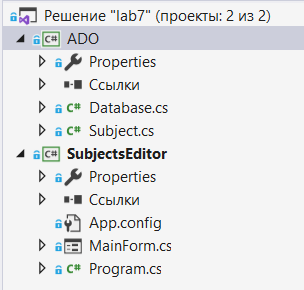


Рисунок 2 – Структура решения

На рисунке 3 изображена схема базы данных.

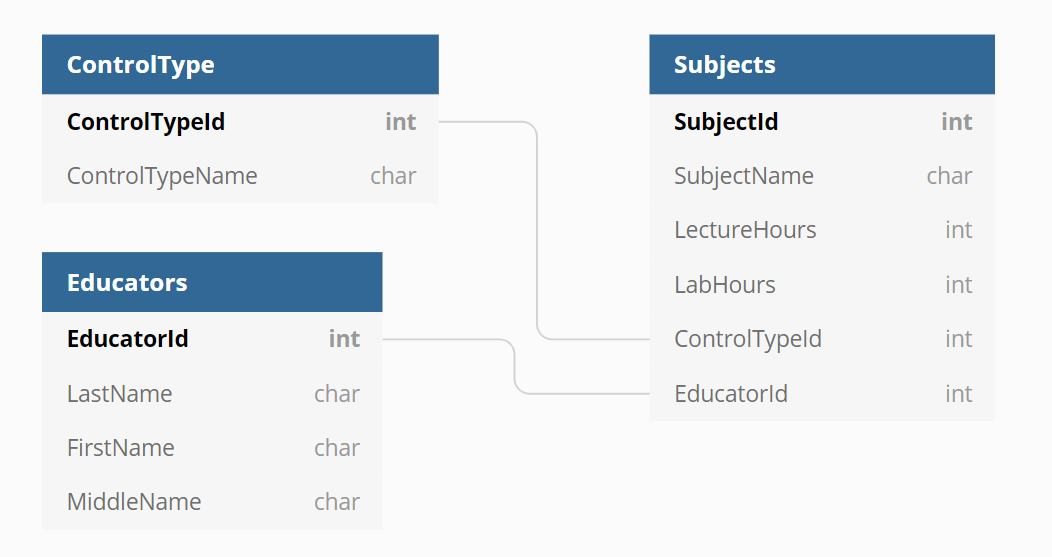


Рисунок 3 – Схема базы данных

При запуске программы открывается основное меню (Рисунок 4).

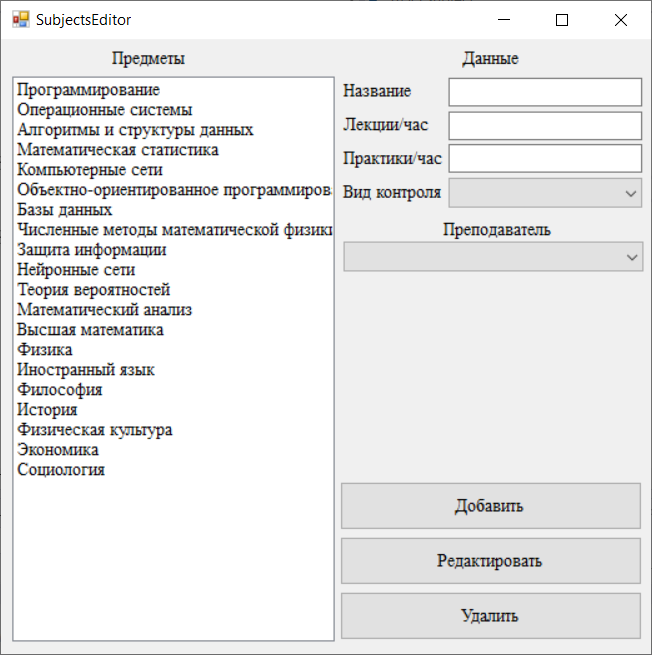


Рисунок 4 – Вывод основного меню

При нажатии кнопки добавления, добавляется новая запись в программе и БД (Рисунок 5).

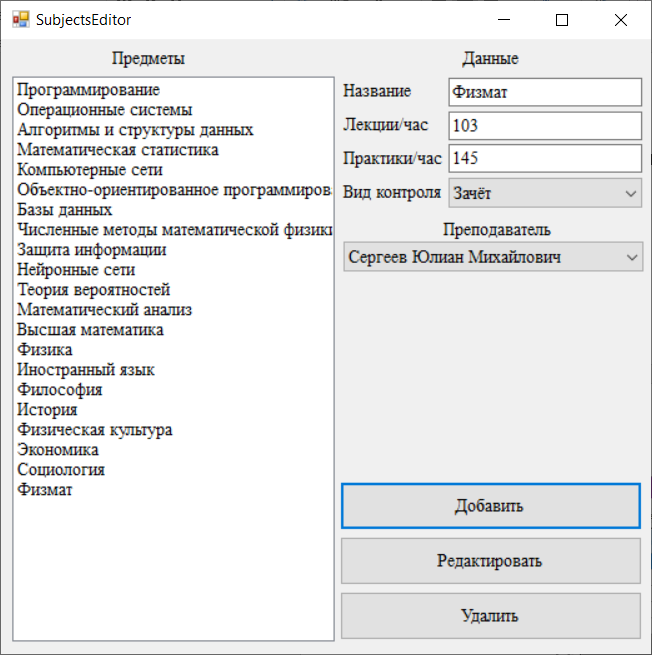


Рисунок 5 – Результат добавления записи

При нажатии кнопки удаления, выбранная запись удаляется из БД (Рисунок 6).

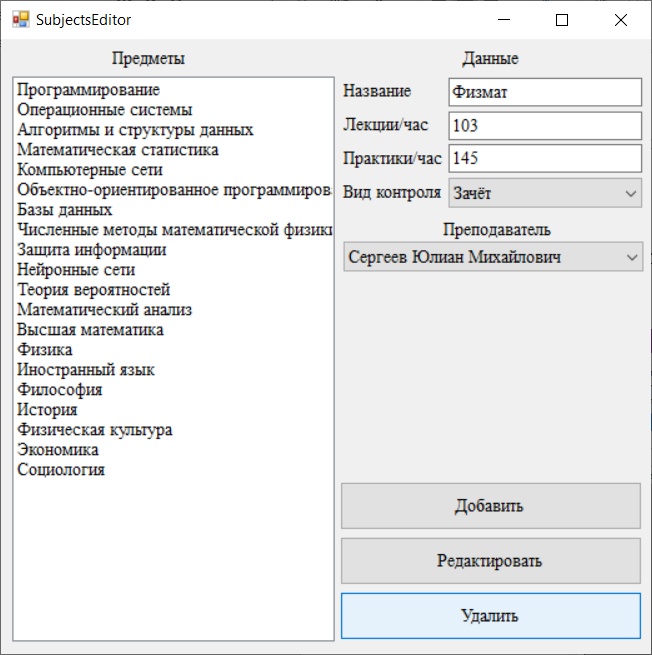


Рисунок 6 – Результат удаления элемента записи

**Вывод:** изучена технология доступа к базам данным.

**ПРИЛОЖЕНИЕ А**

Листинг класса *Database*.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

namespace ADO

{

public class Database

{

static string connectionString = @"Data Source=.\SQLEXPRESS;Initial Catalog=SubjectsDB;Integrated Security=True";

Dictionary<int, string> idControlTypes = new Dictionary<int, string>();

Dictionary<int, string[]> idEducators = new Dictionary<int, string[]>();

public Subject[] subjects;

public Database()

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

//Добавление записей из справочной таблицы в словарь типов контроля

connection.Open();

SqlCommand getControlTypesId = new SqlCommand("select ControlTypeId, ControlTypeName from ControlTypes", connection);

SqlDataReader reader = getControlTypesId.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

idControlTypes.Add(reader.GetInt32(0), reader.GetString(1).TrimEnd(' '));

}

}

reader.Close();

//Добавление записей из справочной таблицы в словарь преподавателей

SqlCommand getEducatorsId = new SqlCommand("select EducatorId, LastName, FirstName, MiddleName from Educators", connection);

reader = getEducatorsId.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

idEducators.Add(reader.GetInt32(0), new string[] { reader.GetString(1).TrimEnd(' '), reader.GetString(2).TrimEnd(' '), reader.GetString(3).TrimEnd(' ') });

}

}

reader.Close();

}

GetSubjects();

}

public string[] GetControlTypes()

{

string[] controlTypes = new string[idControlTypes.Count];

int i = 0;

foreach (KeyValuePair<int, string> idControlType in idControlTypes)

{

controlTypes[i] = idControlType.Value;

i += 1;

}

return controlTypes;

}

public string[] GetEducators()

{

string[] educators = new string[idEducators.Count];

int i = 0;

foreach (KeyValuePair<int, string[]> idEducator in idEducators)

{

educators[i] = idEducator.Value[0] + " " +

idEducator.Value[1] + " " +

idEducator.Value[2];

i += 1;

}

return educators;

}

public void GetSubjects()

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

//Добавление записей из справочной таблицы в словарь типов контроля

connection.Open();

string sqlCom = "select S.SubjectName, S.LectureHours, S.LabHours, C.ControlTypeName, " +

"E.LastName, E.FirstName, E.MiddleName " +

"from Subjects as S join ControlTypes as C " +

"on S.ControlTypeId = C.ControlTypeId join Educators as E " +

"on S.EducatorId = E.EducatorId";

SqlCommand getSubjects = new SqlCommand(sqlCom, connection);

SqlDataReader reader = getSubjects.ExecuteReader();

List<Subject> subjects = new List<Subject>();

if (reader.HasRows)

{

while (reader.Read())

{

subjects.Add(new Subject

{

Name = reader.GetString(0).TrimEnd(' '),

lecHours = reader.GetInt32(1),

labHours = reader.GetInt32(2),

ControlType = reader.GetString(3).TrimEnd(' '),

Educator = reader.GetString(4).TrimEnd(' ') + " " +

reader.GetString(5).TrimEnd(' ') + " " +

reader.GetString(6).TrimEnd(' ')

});

}

}

reader.Close();

this.subjects = subjects.ToArray();

}

}

public void AddSubject(Subject subject)

{

if (subject.Name == "" || subject.Educator == "" || subject.ControlType == "" || subject.lecHours > 200

|| subject.lecHours < 0 || subject.labHours > 200 || subject.labHours < 0)

{

throw new Exception("Вы ввели неверные данные.");

}

foreach (Subject sub in subjects)

{

if (sub.Name.Equals(subject.Name))

throw new Exception("Предмет с таким названием уже есть в базе данных");

}

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand("insert into Subjects(SubjectName, LectureHours, LabHours, ControlTypeId, EducatorId) values(@name, @lech, @labh, @contrid, @educid)", connection);

SqlParameter nameParam = new SqlParameter("@name", subject.Name);

SqlParameter lechParam = new SqlParameter("@lech", subject.lecHours);

SqlParameter labhParam = new SqlParameter("@labh", subject.labHours);

SqlParameter contridParam = new SqlParameter("@contrid", FindControlIdByName(subject.ControlType));

SqlParameter educidParam = new SqlParameter("@educid", FindEducatorByName(subject.Educator));

command.Parameters.Add(nameParam);

command.Parameters.Add(lechParam);

command.Parameters.Add(labhParam);

command.Parameters.Add(contridParam);

command.Parameters.Add(educidParam);

command.ExecuteNonQuery();

}

}

public void EditSubject(Subject subject)

{

if (subject.Name == "" || subject.Educator == "" || subject.ControlType == "" || subject.lecHours > 200

|| subject.lecHours < 0 || subject.labHours > 200 || subject.labHours < 0)

{

throw new Exception("Вы ввели неверные данные.");

}

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand("update Subjects set LectureHours = @lech, LabHours = @labh, " +

"ControlTypeId = @contrid, EducatorId = @educid " +

"where SubjectName = @name", connection);

SqlParameter nameParam = new SqlParameter("@name", subject.Name);

SqlParameter lechParam = new SqlParameter("@lech", subject.lecHours);

SqlParameter labhParam = new SqlParameter("@labh", subject.labHours);

SqlParameter contridParam = new SqlParameter("@contrid", FindControlIdByName(subject.ControlType));

SqlParameter educidParam = new SqlParameter("@educid", FindEducatorByName(subject.Educator));

command.Parameters.Add(nameParam);

command.Parameters.Add(lechParam);

command.Parameters.Add(labhParam);

command.Parameters.Add(contridParam);

command.Parameters.Add(educidParam);

command.ExecuteNonQuery();

}

}

public void DeleteSubject(string name)

{

using (SqlConnection connection = new SqlConnection(connectionString))

{

connection.Open();

SqlCommand command = new SqlCommand("delete from Subjects where SubjectName = @name", connection);

SqlParameter nameParam = new SqlParameter("@name", name);

command.Parameters.Add(nameParam);

command.ExecuteNonQuery();

}

}

int FindControlIdByName(string name)

{

foreach (KeyValuePair<int, string> pair in idControlTypes)

{

if (pair.Value == name)

{

return pair.Key;

}

}

throw new Exception("Нет такого значения типа контроля");

}

int FindEducatorByName(string educator)

{

foreach (KeyValuePair<int, string[]> pair in idEducators)

{

if (educator.Equals(pair.Value[0] + " " + pair.Value[1] + " " + pair.Value[2]))

{

return pair.Key;

}

}

throw new Exception("Нет такого преподавателя");

}

}

}

**ПРИЛОЖЕНИЕ Б**

Листинг класса *MainForm.cs*.

using System;

using ADO;

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;

namespace SubjectsEditor

{

public partial class MainForm : Form

{

Database database;

public MainForm()

{

InitializeComponent();

database = new Database();

InitializeControlBox();

InitializeEducators();

InitializeSubjectsList();

}

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

{

try

{

Subject subject = new Subject

{

Name = subjectNameBox.Text,

lecHours = int.Parse(lecHoursBox.Text),

labHours = int.Parse(labHoursBox.Text),

ControlType = controlTypesBox.Text,

Educator = educatorBox.Text

};

database.AddSubject(subject);

InitializeSubjectsList();

}

catch(Exception ex)

{

MessageBox.Show(ex.Message);

}

}

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

{

try

{

Subject subject = new Subject

{

Name = subjectNameBox.Text,

lecHours = int.Parse(lecHoursBox.Text),

labHours = int.Parse(labHoursBox.Text),

ControlType = controlTypesBox.Text,

Educator = educatorBox.Text

};

database.EditSubject(subject);

InitializeSubjectsList();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

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

{

try

{

database.DeleteSubject(subjectsBox.SelectedItem.ToString());

InitializeSubjectsList();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

void InitializeControlBox()

{

controlTypesBox.Items.AddRange(database.GetControlTypes());

}

void InitializeEducators()

{

educatorBox.Items.AddRange(database.GetEducators());

}

void InitializeSubjectsList()

{

subjectsBox.Items.Clear();

database.GetSubjects();

foreach(Subject sub in database.subjects)

{

subjectsBox.Items.Add(sub.Name);

}

}

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

{

int idx = subjectsBox.SelectedIndex;

subjectNameBox.Text = database.subjects[idx].Name;

lecHoursBox.Text = database.subjects[idx].lecHours.ToString();

labHoursBox.Text = database.subjects[idx].labHours.ToString();

controlTypesBox.SelectedItem = database.subjects[idx].ControlType;

educatorBox.SelectedItem = database.subjects[idx].Educator;

}

}

}