Лабораторная работа № 4.  
Работа с Хранилищами данных,

Entity Framework

**Цель лабораторной работы**

Закрепление теоретических знаний по основам работы с баззой данных при помощи технологии Entity Framework.

**Постановка задачи**

**ADO.NET Entity Framework** — объектно-ориентированная технология доступа к данным, является object-relational mapping (отображение объектов и связей) решением для .NET Framework от Microsoft. Предоставляет возможность взаимодействия с объектами как посредством LINQ в виде LINQ to Entities, так и с использованием Entity SQL.

Для работы с БД будем использовать СУБД SQLite и EntityFramework. Необходимо перейти в диспетчер пакетов Nuget и установить следующие пакеты (рисунок 1):

* Microsoft.EntityFrameworkCore,
* Microsoft.EntityFrameworkCore.Sqlite.

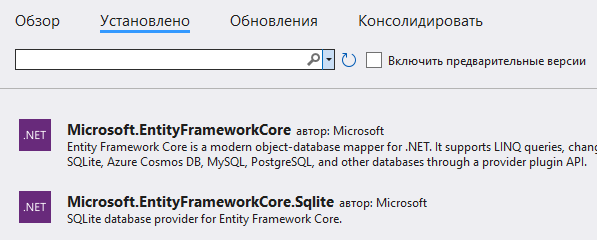


Рисунок 1 – Установка пакетов для работы с базой данных

**Ба́за да́нных** — совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняют в соответствии с правилами средств моделирования данных.

**Систе́ма управле́ния ба́зами да́нных, сокр. СУБД** — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных. СУБД — комплекс программ, позволяющих создать базу данных и манипулировать данными.

**Таблица** — совокупность связанных данных, хранящихся в структурированном виде в базе данных. Она состоит из столбцов и строк. Таблица содержит определенное число столбцов, но может иметь любое количество строк. Каждая строка однозначно определяется одним или несколькими уникальными значениями, которые принимают её ячейки из определенного подмножества столбцов. Подмножество столбцов, которое уникально идентифицирует строку, называется первичным ключом.

**Ячейка** — место, где строка и столбец пересекаются.

**Первичный ключ** – поле, которое уникально идентифицирует запись в таблице. Обычно в качестве ключа используется тип int и имя Id.

Чтобы задать имя таблицы используется атрибут [Table(“Имя”)].

Чтобы явным образом указать первичный ключ нужно использовать атрибут [Key].

Чтобы задать минимальную или максимальную длину строки используются атрибуты [MinLength(N)] и [MaxLength(N)].

[Table("Покупатель")]

public class Customer

{

[Key] public int Id { get; set; }

[MaxLength(200)] public string? Name { get; set; }

public int Age { get; set; }

}

Добавим класс банковского аккаунта пользователя:

public class BankAccount

{

public int Id { get; set; }

public int CustomerId { get; set; } //Внешний ключ

public double Money { get; set; }

public Customer? Customer { get; set; } //Навигационное свойство

}

Добавим класс, описывающий товар:

public class Product

{

public int Id { get; set; }

public string? Name { get; set; }

public double Price { get; set; }

}

Взаимодействие с базой данных в **Entity Framework Core** происходит посредством специального класса - **контекста данных**. Поэтому добавим в наш проект новый класс, который назовем ApplicationContext и который будет иметь следующий код:

public class ApplicationContext : DbContext

{

public DbSet<Customer> Customers => Set<Customer>();

public DbSet<Product> Products => Set<Product>();

public DbSet<BankAccount> BankAccounts => Set<BankAccount>();

public ApplicationContext() => Database.EnsureCreated();

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

optionsBuilder.UseSqlite("Data Source=MyDataBase.db");

}

}

Посмотрим на примерах как можно работать с таблицами, заполним БД некоторыми данными:

using ApplicationContext db = new();

db.Customers.AddRange(new List<Customer> {

new Customer{Name = "Иван", Age = 25},

new Customer{Name = "Петр", Age = 35},

new Customer{Name = "Елена", Age = 23},

new Customer{Name = "Анна", Age = 40},

});

db.Products.AddRange(new List<Product> {

new Product{Name = "Стол", Price = 12000},

new Product{Name = "Стул", Price = 2500},

new Product{Name = "ВАЗ 2107", Price = 75000},

new Product{Name = "Яблоки 1кг", Price = 120},

});

db.SaveChanges();

**Примеры** работы с БД.

Извлечение данных:

var customers = db.Customers.ToList();

Извлекаем из БД молодых покупателей:

var youngCustomers = db.Customers

.Where(c => c.Age <= 30)

.ToList();

Извлекаем самый дорогой товар:

var mostExpensive = db.Products

.ToList()

.MaxBy(p => p.Price);

Удаляем аккаунты с количеством денег меньше 10000:

var customersToRemove = db.BankAccounts

.Where(b => b.Money <= 1e4)

.Select(b => b.Customer)

.ToList();

foreach (var customer in customersToRemove)

{

db.Customers.Remove(customer);

}

Сохраняем изменения:

db.SaveChanges();

Дополнительная информация по работе с Entity Framework: <https://metanit.com/sharp/efcore/1.2.php>

**Пример** создания загрузчика данных в соответствии с принципами SOLID.

// Класс человека в сокращенном виде

public record Person(int Id, string Name, int Age, double Money);

// Интерфейс для загрузки объекта человека

public interface IPersonLoader

{

public Person? LoadByName(string dataBasePath, string name);

}

using Newtonsoft.Json.Linq;

// Класс для загрузки человека из JSON файла

public class JsonPersonLoader : IPersonLoader

{

public Person? LoadByName(string dataBasePath, string name)

{

var data = File.ReadAllText(dataBasePath);

var jsonArray = JArray.Parse(data);

foreach (var item in jsonArray)

{

var tokenName = item.Value<string>("Name");

if (tokenName == name)

{

var tokenAge = item.Value<int>("Age");

var tokenMoney = item.Value<double>("Money");

return new Person(default, tokenName, tokenAge, tokenMoney);

}

}

return null;

}

}

// Класс для загрузки человека из БД SQLite

public class SQLitePersonLoader : IPersonLoader

{

public Person? LoadByName(string dataBasePath, string name)

{

using ApplicationDBContext db = new(dataBasePath);

return db.Persons.SingleOrDefault(p => p.Name == name);

}

}

// Интерфейс для сохранения объекта человека

public interface IPersonSaver

{

public void SavePerson(string dataBasePath, Person person);

}

using Newtonsoft.Json.Linq;

// Класс для сохранения человека в JSON

public class JsonPersonSaver : IPersonSaver

{

public void SavePerson(string dataBasePath, Person person)

{

var json = File.ReadAllText(dataBasePath);

var jArr = JArray.Parse(json);

var jsonPerson = new JObject();

jsonPerson["Name"] = person.Name;

jsonPerson["Age"] = person.Age;

jsonPerson["Money"] = person.Money;

jArr.Add(jsonPerson);

File.WriteAllText(dataBasePath, jArr.ToString());

}

}

// Класс для сохранения человека в БД SQLite

public class SQLitePersonSaver : IPersonSaver

{

public void SavePerson(string dataBasePath, Person person)

{

using ApplicationDBContext db = new(dataBasePath);

db.Persons.Add(person);

db.SaveChanges();

}

}

**Задание на лабораторную работу**

1. Необходимо разработать информационную систему с графическим интерфейсом в WindowsForms.
2. Программа должна быть загружена в GitHub репозиторий.
3. Предусмотреть формы для CRUD-операций (Create, Read, Update, Delete).
4. Предусмотреть валидацию ввода (проверка на пустые поля).
5. Предусмотреть фильтрацию данных (например, поиск книги по автору).
6. **Для оценки «Удовлетворительно».** Использовать 1 класс.
7. **Для оценки «Хорошо».** Использовать 2 класса.
8. **Для оценки «Отлично».** Использовать 3 класса.
9. Защита работы включает демонстрацию работы программы.
10. Для успешной защиты:

* программа должна быть задокументирована с помощью комментариев,
* код программы не должен содержать не используемые блоки, должен быть «чистым» и стилистически верно оформлен,
* интерфейс программы должен быть гибким и удобным.

1. Библиотека книг

* Классы: Book, Author, Genre.

1. Музыкальная коллекция

* Классы: Song, Artist, Album.

1. Фильмотека

* Классы: Movie, Director, Studio.

1. Журналы и статьи

* Классы: Article, Journal, Category.

1. Аудиокниги

* Классы: Audiobook, Narrator, Publisher.

1. Комиксы

* Классы: Comic, Writer, Series.

1. Видеоигры

* Классы: Game, Developer, Platform.

1. Научные работы

* Классы: ResearchPaper, Scientist, Field.

1. Фотогалерея

* Классы: Photo, Photographer, Location.

1. Коллекция рецептов

* Классы: Recipe, Chef, Cuisine.

1. Учёт студентов

* Классы: Student, Group, Faculty.

1. Расписание занятий

* Классы: Lesson, Teacher, Classroom.

1. Спортивные секции

* Классы: Athlete, Coach, Sport.

1. Школьные предметы

* Классы: Subject, Teacher, Grade.

1. Курсы онлайн-обучения

* Классы: Course, Instructor, Topic.

1. Турниры по играм

* Классы: Tournament, Player, Game.

1. Фитнес-клуб

* Классы: Client, Trainer, Membership.

1. Музыкальная школа

* Классы: Student, Instrument, Teacher.

1. Библиотека учебников

* Классы: Textbook, Subject, Publisher.

1. Шахматный клуб

* Классы: Player, Tournament, Club.

1. Учёт сотрудников

* Классы: Employee, Department, Position.

1. Заказы в ресторане

* Классы: Order, Dish, Waiter.

1. Автосервис

* Классы: Car, Owner, Service.

1. Аптека

* Классы: Medicine, Manufacturer, Category.

1. Гостиница

* Классы: Guest, Room, Booking.

1. Таксопарк

* Классы: Driver, Car, Order.

1. Агентство недвижимости

* Классы: Property, Owner, Agent.

1. Фитнес-трекер

* Классы: Workout, User, Exercise.

1. Салон красоты

* Классы: Client, Service, Master.

1. Фермерское хозяйство

* Классы: Animal, Breed, Farm.

**Пример разработки приложения**

Класс: Animal

**Шаг 1: Создание проекта в Visual Studio 2022**

1. Откройте Visual Studio 2022
2. Создайте новый проект:
   * Выберите "Windows Forms App" (используйте поиск по шаблонам)
3. Назовите проект: AnimalInformationSystem
4. В выпадающем списке выберите .NET 8.0
5. Нажмите "Создать"

**Шаг 2: Установка необходимых пакетов NuGet**

1. Правой кнопкой по проекту → "Управление пакетами NuGet"
2. Установите следующие пакеты:
   * Microsoft.EntityFrameworkCore.Sqlite
   * Microsoft.EntityFrameworkCore.Design
   * Microsoft.EntityFrameworkCore.Tools

**Шаг 3: Создание модели данных (Code First)**

Добавьте класс Animal.cs:

using System.ComponentModel.DataAnnotations;

namespace AnimalInformationSystem;

public class Animal

{

[Key]

public int Id { get; set; }

[Required(ErrorMessage = "Имя обязательно")]

[StringLength(100, ErrorMessage = "Имя не должно превышать 100 символов")]

public string Name { get; set; } = string.Empty;

[Required(ErrorMessage = "Вид обязателен")]

[StringLength(100, ErrorMessage = "Вид не должен превышать 100 символов")]

public string Species { get; set; } = string.Empty;

[Required(ErrorMessage = "Возраст обязателен")]

[Range(0, 100, ErrorMessage = "Возраст должен быть между 0 и 100")]

public int Age { get; set; }

[Required(ErrorMessage = "Вес обязателен")]

[Range(0.1, 1000, ErrorMessage = "Вес должен быть между 0.1 и 1000")]

public double Weight { get; set; }

[StringLength(500, ErrorMessage = "Описание не должно превышать 500 символов")]

public string Description { get; set; } = string.Empty;

[Required(ErrorMessage = "Дата поступления обязательна")]

public DateTime AdmissionDate { get; set; } = DateTime.Now;

}

**Шаг 4: Создание контекста базы данных**

Добавьте класс AnimalContext.cs:

using Microsoft.EntityFrameworkCore;

namespace AnimalInformationSystem;

public class AnimalContext : DbContext

{

public DbSet<Animal> Animals { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

optionsBuilder.UseSqlite("Data Source=animals.db");

}

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<Animal>().ToTable("Animals");

}

}

**Шаг 5: Настройка главной формы (MainForm, рисунок 2)**

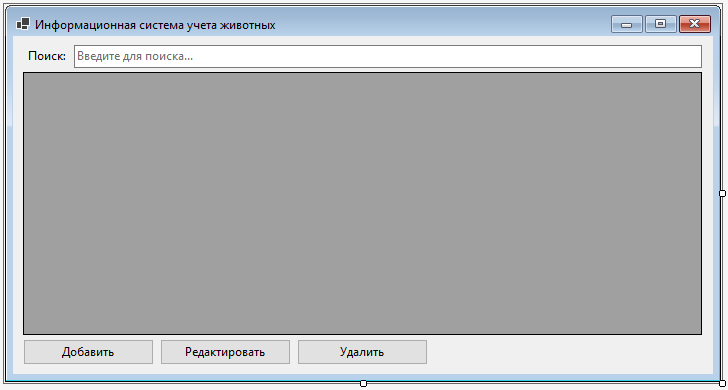


Рисунок 2 – Главное окно программы

1. Откройте MainForm.cs в конструкторе
2. Добавьте следующие элементы:
   * DataGridView (dgvAnimals)
   * TextBox для поиска (txtSearch)
   * Кнопки: "Добавить", "Редактировать", "Удалить"
3. Обновите код формы:

namespace AnimalInformationSystem;

public partial class MainForm : Form

{

private readonly AnimalContext \_db = new();

public MainForm()

{

InitializeComponent();

\_db.Database.EnsureCreated();

LoadAnimals();

}

private void LoadAnimals(string filter = "")

{

var query = \_db.Animals.AsQueryable();

if (!string.IsNullOrEmpty(filter))

{

query = query.Where(a => a.Name.Contains(filter) ||

a.Species.Contains(filter));

}

dgvAnimals.DataSource = query.ToList();

dgvAnimals.Columns["Id"].Visible = false;

}

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

{

using var form = new EditAnimalForm();

if (form.ShowDialog() == DialogResult.OK)

{

\_db.Animals.Add(form.Animal);

\_db.SaveChanges();

LoadAnimals();

}

}

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

{

if (dgvAnimals.SelectedRows.Count == 0) return;

var id = (int)dgvAnimals.SelectedRows[0].Cells["Id"].Value;

var animal = \_db.Animals.Find(id);

if (animal == null) return;

using var form = new EditAnimalForm(animal);

if (form.ShowDialog() == DialogResult.OK)

{

\_db.SaveChanges();

LoadAnimals();

}

}

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

{

if (dgvAnimals.SelectedRows.Count == 0) return;

var id = (int)dgvAnimals.SelectedRows[0].Cells["Id"].Value;

var animal = \_db.Animals.Find(id);

if (animal == null) return;

if (MessageBox.Show("Удалить запись?", "Подтверждение",

MessageBoxButtons.YesNo) == DialogResult.Yes)

{

\_db.Animals.Remove(animal);

\_db.SaveChanges();

LoadAnimals();

}

}

private void txtSearch\_TextChanged(object sender, EventArgs e)

{

LoadAnimals(txtSearch.Text);

}

}

**Шаг 6: Создание формы редактирования (рисунок 3)**

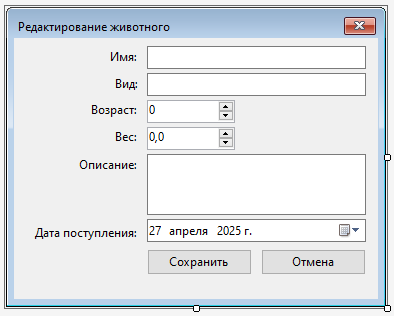


Рисунок 3 – Окно редактирования животного

1. Добавьте новую форму (EditAnimalForm.cs)
2. Разместите элементы управления для редактирования всех свойств Animal
3. Добавьте код:

namespace AnimalInformationSystem;

public partial class EditAnimalForm : Form

{

public Animal Animal { get; }

public EditAnimalForm(Animal? animal = null)

{

InitializeComponent();

Animal = animal ?? new Animal();

SetupDataBindings();

}

private void SetupDataBindings()

{

txtName.DataBindings.Add("Text", Animal, "Name", false, DataSourceUpdateMode.OnPropertyChanged);

txtSpecies.DataBindings.Add("Text", Animal, "Species", false, DataSourceUpdateMode.OnPropertyChanged);

numAge.DataBindings.Add("Value", Animal, "Age", false, DataSourceUpdateMode.OnPropertyChanged);

numWeight.DataBindings.Add("Value", Animal, "Weight", false, DataSourceUpdateMode.OnPropertyChanged);

txtDescription.DataBindings.Add("Text", Animal, "Description", false, DataSourceUpdateMode.OnPropertyChanged);

dtpAdmissionDate.DataBindings.Add("Value", Animal, "AdmissionDate", false, DataSourceUpdateMode.OnPropertyChanged);

}

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

{

if (ValidateChildren())

{

DialogResult = DialogResult.OK;

Close();

}

}

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

{

DialogResult = DialogResult.Cancel;

Close();

}

private void txtName\_Validating(object sender, System.ComponentModel.CancelEventArgs e)

{

if (string.IsNullOrWhiteSpace(txtName.Text))

{

errorProvider.SetError(txtName, "Имя обязательно");

e.Cancel = true;

}

else

{

errorProvider.SetError(txtName, "");

}

}

private void txtSpecies\_Validating(object sender, System.ComponentModel.CancelEventArgs e)

{

if (string.IsNullOrWhiteSpace(txtSpecies.Text))

{

errorProvider.SetError(txtSpecies, "Вид обязателен");

e.Cancel = true;

}

else

{

errorProvider.SetError(txtSpecies, "");

}

}

}

**Шаг 7: Настройка валидации**

1. Для всех TextBox установите:
   * CausesValidation = true
2. Для NumericUpDown установите соответствующие Minimum/Maximum значения
3. Добавьте ErrorProvider на форму

**Шаг 8: Запуск приложения**

1. Нажмите F5 для запуска
2. Проверьте все функции:
   * Добавление/редактирование/удаление записей
   * Поиск по имени и виду
   * Валидацию данных

Код из дизайнера:

**MainForm.Designer.cs**

namespace AnimalInformationSystem

{

partial class MainForm

{

private System.ComponentModel.IContainer components = null;

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

private void InitializeComponent()

{

dgvAnimals = new DataGridView();

txtSearch = new TextBox();

btnAdd = new Button();

btnEdit = new Button();

btnDelete = new Button();

label1 = new Label();

((System.ComponentModel.ISupportInitialize)dgvAnimals).BeginInit();

SuspendLayout();

//

// dgvAnimals

//

dgvAnimals.AllowUserToAddRows = false;

dgvAnimals.AllowUserToDeleteRows = false;

dgvAnimals.Anchor = AnchorStyles.Top | AnchorStyles.Left | AnchorStyles.Right;

dgvAnimals.ColumnHeadersHeightSizeMode = DataGridViewColumnHeadersHeightSizeMode.AutoSize;

dgvAnimals.Location = new Point(10, 36);

dgvAnimals.Margin = new Padding(3, 2, 3, 2);

dgvAnimals.MultiSelect = false;

dgvAnimals.Name = "dgvAnimals";

dgvAnimals.ReadOnly = true;

dgvAnimals.RowHeadersWidth = 51;

dgvAnimals.SelectionMode = DataGridViewSelectionMode.FullRowSelect;

dgvAnimals.Size = new Size(679, 263);

dgvAnimals.TabIndex = 0;

//

// txtSearch

//

txtSearch.Anchor = AnchorStyles.Top | AnchorStyles.Left | AnchorStyles.Right;

txtSearch.Location = new Point(61, 9);

txtSearch.Margin = new Padding(3, 2, 3, 2);

txtSearch.Name = "txtSearch";

txtSearch.PlaceholderText = "Введите для поиска...";

txtSearch.Size = new Size(628, 23);

txtSearch.TabIndex = 1;

txtSearch.TextChanged += txtSearch\_TextChanged;

//

// btnAdd

//

btnAdd.Anchor = AnchorStyles.Bottom | AnchorStyles.Left;

btnAdd.Location = new Point(10, 303);

btnAdd.Margin = new Padding(3, 2, 3, 2);

btnAdd.Name = "btnAdd";

btnAdd.Size = new Size(131, 26);

btnAdd.TabIndex = 2;

btnAdd.Text = "Добавить";

btnAdd.UseVisualStyleBackColor = true;

btnAdd.Click += btnAdd\_Click;

//

// btnEdit

//

btnEdit.Anchor = AnchorStyles.Bottom | AnchorStyles.Left;

btnEdit.Location = new Point(147, 303);

btnEdit.Margin = new Padding(3, 2, 3, 2);

btnEdit.Name = "btnEdit";

btnEdit.Size = new Size(131, 26);

btnEdit.TabIndex = 3;

btnEdit.Text = "Редактировать";

btnEdit.UseVisualStyleBackColor = true;

btnEdit.Click += btnEdit\_Click;

//

// btnDelete

//

btnDelete.Anchor = AnchorStyles.Bottom | AnchorStyles.Left;

btnDelete.Location = new Point(284, 303);

btnDelete.Margin = new Padding(3, 2, 3, 2);

btnDelete.Name = "btnDelete";

btnDelete.Size = new Size(131, 26);

btnDelete.TabIndex = 4;

btnDelete.Text = "Удалить";

btnDelete.UseVisualStyleBackColor = true;

btnDelete.Click += btnDelete\_Click;

//

// label1

//

label1.AutoSize = true;

label1.Location = new Point(12, 12);

label1.Name = "label1";

label1.Size = new Size(45, 15);

label1.TabIndex = 5;

label1.Text = "Поиск:";

//

// MainForm

//

AutoScaleDimensions = new SizeF(7F, 15F);

AutoScaleMode = AutoScaleMode.Font;

ClientSize = new Size(700, 338);

Controls.Add(label1);

Controls.Add(btnDelete);

Controls.Add(btnEdit);

Controls.Add(btnAdd);

Controls.Add(txtSearch);

Controls.Add(dgvAnimals);

Margin = new Padding(3, 2, 3, 2);

MinimumSize = new Size(527, 310);

Name = "MainForm";

StartPosition = FormStartPosition.CenterScreen;

Text = "Информационная система учета животных";

((System.ComponentModel.ISupportInitialize)dgvAnimals).EndInit();

ResumeLayout(false);

PerformLayout();

}

#endregion

private DataGridView dgvAnimals;

private TextBox txtSearch;

private Button btnAdd;

private Button btnEdit;

private Button btnDelete;

private Label label1;

}

}

**EditAnimalForm.Designer.cs**

namespace AnimalInformationSystem

{

partial class EditAnimalForm

{

private System.ComponentModel.IContainer components = null;

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

private void InitializeComponent()

{

components = new System.ComponentModel.Container();

label2 = new Label();

txtName = new TextBox();

label3 = new Label();

txtSpecies = new TextBox();

label4 = new Label();

numAge = new NumericUpDown();

label5 = new Label();

numWeight = new NumericUpDown();

label6 = new Label();

txtDescription = new TextBox();

label7 = new Label();

dtpAdmissionDate = new DateTimePicker();

btnSave = new Button();

btnCancel = new Button();

errorProvider = new ErrorProvider(components);

((System.ComponentModel.ISupportInitialize)numAge).BeginInit();

((System.ComponentModel.ISupportInitialize)numWeight).BeginInit();

((System.ComponentModel.ISupportInitialize)errorProvider).BeginInit();

SuspendLayout();

//

// label2

//

label2.Anchor = AnchorStyles.Top | AnchorStyles.Right;

label2.AutoSize = true;

label2.Location = new Point(82, 14);

label2.Name = "label2";

label2.Size = new Size(34, 15);

label2.TabIndex = 0;

label2.Text = "Имя:";

//

// txtName

//

txtName.Anchor = AnchorStyles.Top | AnchorStyles.Right;

txtName.Location = new Point(122, 11);

txtName.Margin = new Padding(3, 2, 3, 2);

txtName.Name = "txtName";

txtName.Size = new Size(219, 23);

txtName.TabIndex = 1;

txtName.Validating += txtName\_Validating;

//

// label3

//

label3.Anchor = AnchorStyles.Top | AnchorStyles.Right;

label3.AutoSize = true;

label3.Location = new Point(87, 41);

label3.Name = "label3";

label3.Size = new Size(30, 15);

label3.TabIndex = 2;

label3.Text = "Вид:";

//

// txtSpecies

//

txtSpecies.Anchor = AnchorStyles.Top | AnchorStyles.Right;

txtSpecies.Location = new Point(122, 38);

txtSpecies.Margin = new Padding(3, 2, 3, 2);

txtSpecies.Name = "txtSpecies";

txtSpecies.Size = new Size(219, 23);

txtSpecies.TabIndex = 3;

txtSpecies.Validating += txtSpecies\_Validating;

//

// label4

//

label4.Anchor = AnchorStyles.Top | AnchorStyles.Right;

label4.AutoSize = true;

label4.Location = new Point(63, 67);

label4.Name = "label4";

label4.Size = new Size(53, 15);

label4.TabIndex = 4;

label4.Text = "Возраст:";

//

// numAge

//

numAge.Anchor = AnchorStyles.Top | AnchorStyles.Right;

numAge.Location = new Point(122, 65);

numAge.Margin = new Padding(3, 2, 3, 2);

numAge.Name = "numAge";

numAge.Size = new Size(88, 23);

numAge.TabIndex = 5;

//

// label5

//

label5.Anchor = AnchorStyles.Top | AnchorStyles.Right;

label5.AutoSize = true;

label5.Location = new Point(87, 94);

label5.Name = "label5";

label5.Size = new Size(29, 15);

label5.TabIndex = 6;

label5.Text = "Вес:";

//

// numWeight

//

numWeight.Anchor = AnchorStyles.Top | AnchorStyles.Right;

numWeight.DecimalPlaces = 1;

numWeight.Increment = new decimal(new int[] { 1, 0, 0, 65536 });

numWeight.Location = new Point(122, 92);

numWeight.Margin = new Padding(3, 2, 3, 2);

numWeight.Maximum = new decimal(new int[] { 1000, 0, 0, 0 });

numWeight.Name = "numWeight";

numWeight.Size = new Size(88, 23);

numWeight.TabIndex = 7;

//

// label6

//

label6.Anchor = AnchorStyles.Top | AnchorStyles.Right;

label6.AutoSize = true;

label6.Location = new Point(51, 122);

label6.Name = "label6";

label6.Size = new Size(65, 15);

label6.TabIndex = 8;

label6.Text = "Описание:";

//

// txtDescription

//

txtDescription.Anchor = AnchorStyles.Top | AnchorStyles.Right;

txtDescription.Location = new Point(122, 119);

txtDescription.Margin = new Padding(3, 2, 3, 2);

txtDescription.Multiline = true;

txtDescription.Name = "txtDescription";

txtDescription.Size = new Size(219, 61);

txtDescription.TabIndex = 9;

//

// label7

//

label7.Anchor = AnchorStyles.Top | AnchorStyles.Right;

label7.AutoSize = true;

label7.Location = new Point(7, 190);

label7.Name = "label7";

label7.Size = new Size(109, 15);

label7.TabIndex = 10;

label7.Text = "Дата поступления:";

//

// dtpAdmissionDate

//

dtpAdmissionDate.Anchor = AnchorStyles.Top | AnchorStyles.Right;

dtpAdmissionDate.Location = new Point(122, 184);

dtpAdmissionDate.Margin = new Padding(3, 2, 3, 2);

dtpAdmissionDate.Name = "dtpAdmissionDate";

dtpAdmissionDate.Size = new Size(219, 23);

dtpAdmissionDate.TabIndex = 11;

//

// btnSave

//

btnSave.Anchor = AnchorStyles.Top | AnchorStyles.Right;

btnSave.Location = new Point(122, 214);

btnSave.Margin = new Padding(3, 2, 3, 2);

btnSave.Name = "btnSave";

btnSave.Size = new Size(105, 26);

btnSave.TabIndex = 12;

btnSave.Text = "Сохранить";

btnSave.UseVisualStyleBackColor = true;

btnSave.Click += btnSave\_Click;

//

// btnCancel

//

btnCancel.Anchor = AnchorStyles.Top | AnchorStyles.Right;

btnCancel.Location = new Point(236, 214);

btnCancel.Margin = new Padding(3, 2, 3, 2);

btnCancel.Name = "btnCancel";

btnCancel.Size = new Size(105, 26);

btnCancel.TabIndex = 13;

btnCancel.Text = "Отмена";

btnCancel.UseVisualStyleBackColor = true;

btnCancel.Click += btnCancel\_Click;

//

// errorProvider

//

errorProvider.ContainerControl = this;

//

// EditAnimalForm

//

AutoScaleDimensions = new SizeF(7F, 15F);

AutoScaleMode = AutoScaleMode.Font;

ClientSize = new Size(381, 261);

Controls.Add(btnCancel);

Controls.Add(btnSave);

Controls.Add(dtpAdmissionDate);

Controls.Add(label7);

Controls.Add(txtDescription);

Controls.Add(label6);

Controls.Add(numWeight);

Controls.Add(label5);

Controls.Add(numAge);

Controls.Add(label4);

Controls.Add(txtSpecies);

Controls.Add(label3);

Controls.Add(txtName);

Controls.Add(label2);

FormBorderStyle = FormBorderStyle.FixedDialog;

Margin = new Padding(3, 2, 3, 2);

MaximizeBox = false;

MinimizeBox = false;

MinimumSize = new Size(380, 300);

Name = "EditAnimalForm";

StartPosition = FormStartPosition.CenterParent;

Text = "Редактирование животного";

((System.ComponentModel.ISupportInitialize)numAge).EndInit();

((System.ComponentModel.ISupportInitialize)numWeight).EndInit();

((System.ComponentModel.ISupportInitialize)errorProvider).EndInit();

ResumeLayout(false);

PerformLayout();

}

#endregion

private Label label2;

private TextBox txtName;

private Label label3;

private TextBox txtSpecies;

private Label label4;

private NumericUpDown numAge;

private Label label5;

private NumericUpDown numWeight;

private Label label6;

private TextBox txtDescription;

private Label label7;

private DateTimePicker dtpAdmissionDate;

private Button btnSave;

private Button btnCancel;

private ErrorProvider errorProvider;

}

}

# Содержание пояснительной записки

1. Постановка задачи. Приводится теоретический материал, использованный при написании приложения.

2. Формулировка задания и вариант. Приводится задание на лабораторную работу и вариант этого задания.

3. Описание выполняемых действий. Необходимо привести описание последовательности разработки программы, реализации используемых методов, алгоритмов, блок-схем.

4. Анализ результатов. Привести анализ входных и выходных данных. Показать результаты выполнения программного кода. Предоставить скриншоты обработки тестовых примеров. Сделать выводы.

5. Листинг программы. Привести листинг разработанного программного кода, содержание файлов входных и выходных данных.

# Используемое программное обеспечение

1. Среда программирования MS Visual Studio Community 2022 (Свободно распространяемое программное обеспечение (в учебных целях));
2. Microsoft Office Standard 2007 (Open License: 42267924);
3. Open Office (Свободно распространяемое программное обеспечение).
4. Браузер (Свободно распространяемое программное обеспечение).

# Список литературы

* + - 1. Мейер Б. Объектно-ориентированное программирование и программная инженерия [Электронный ресурс]/ Мейер Б. – Электрон. текстовые данные. – М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. – 285 c.
      2. Биллиг, В. A. Основы объектного программирования на С# (C# 3.0, Visual Studio 2008) [Электронный ресурс] : учебное пособие / В. A. Биллиг. — Электрон. текстовые данные. — Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017. — 583 c. — 978-5-4487-0145-0. — Режим доступа: http://www.iprbookshop.ru/72339.html
      3. Павловская, Т. А. Программирование на языке высокого уровня C# [Электронный ресурс] / Т. А. Павловская. — 2-е изд. — Электрон. текстовые данные. — М. : Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 245 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/73713.html
      4. Агапов, В. П. Основы программирования на языке С# [Электронный ресурс] : учебное пособие / В. П. Агапов. — Электрон. текстовые данные. — М. : Московский государственный строительный университет, ЭБС АСВ, 2012. — 128 c. — 978-5-7264-0576-6. — Режим доступа: http://www.iprbookshop.ru/16366.html
      5. Медведев, М. А. Программирование на СИ# [Электронный ресурс] : учебное пособие / М. А. Медведев, А. Н. Медведев ; под ред. А. В. Присяжный. — Электрон. текстовые данные. — Екатеринбург : Уральский федеральный университет, ЭБС АСВ, 2015. — 64 c. — 978-5-7996-1561-1. — Режим доступа: http://www.iprbookshop.ru/69667.html
      6. Казанский А.А. Объектно-ориентированное программирование на языке Microsoft Visual С# в среде разработки Microsoft Visual Studio 2008 и .NET Framework. 4.3 [Электронный ресурс]: учебное пособие и практикум/ Казанский А.А.— Электрон. текстовые данные.— М.: Московский государственный строительный университет, ЭБС АСВ, 2011.— 180 c
      7. Уйманова Н.А. Основы объектно-ориентированного программирования [Электронный ресурс]: практикум/ Уйманова Н.А., Таспаева М.Г.— Электрон. текстовые данные.— Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2017.— 156 c.— Режим доступа: http://www.iprbookshop.ru/78808.html.— ЭБС «IPRbooks»
      8. Новиков П.В. Объектно-ориентированное программирование [Электронный ресурс]: учебно-методическое пособие к лабораторным работам/ Новиков П.В.— Электрон. текстовые данные.— Саратов: Вузовское образование, 2017.— 124 c.— Режим доступа: http://www.iprbookshop.ru/64650.html.— ЭБС «IPRbooks»