**Оглавление**

[**Лабораторная работа №1** 2](#_Toc44250697)

[**Лабораторная работа №2** 16](#_Toc44250698)

[**Лабораторная работа №3** 29](#_Toc44250699)

[**Лабораторная работа №4** 51](#_Toc44250700)

[**Лабораторная работа №5** 72](#_Toc44250701)

[**Лабораторная работа №6** 86](#_Toc44250702)

[**Лабораторная работа №7** 117](#_Toc44250703)

# **Лабораторная работа №1**

Вариант 13

*Задание*: Реализовать библиотеку классов. Создать классы, в соответствии с вариантом задания. Классы должен содержать как минимум три свойства и три метода. Между классами должны быть обеспечены связи (агрегация, наследование и т.д.), должен быть выбран базовый класс и зависимые (дочерние). Реализовать консольное приложение для создания и заполнения полей объектов библиотеки классов.

*Результат работы программы*

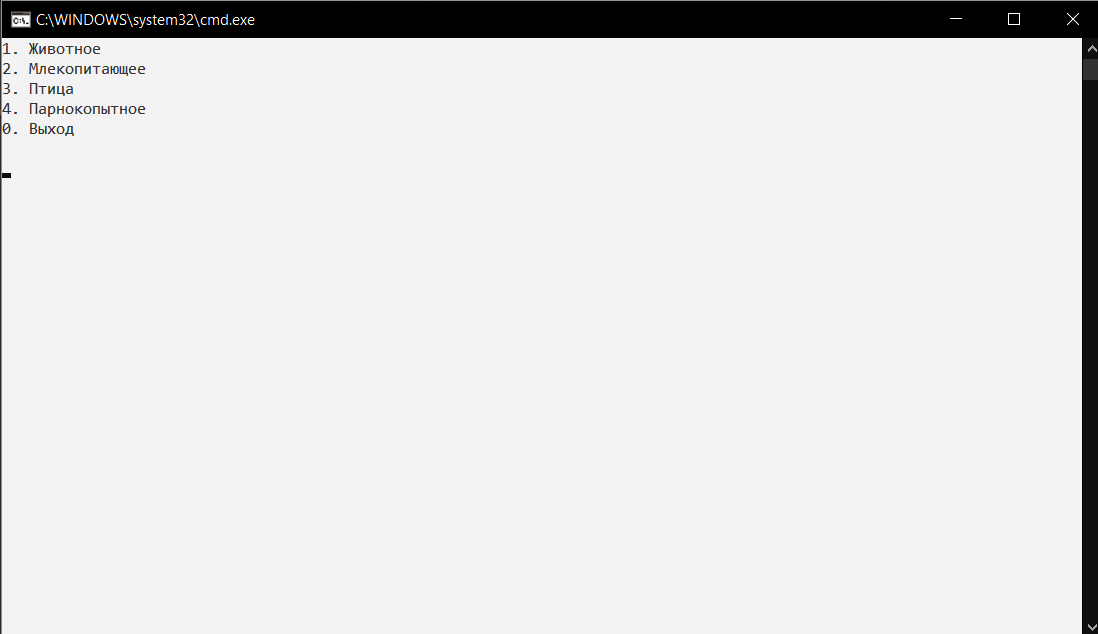


Рис.1 – Главное меню

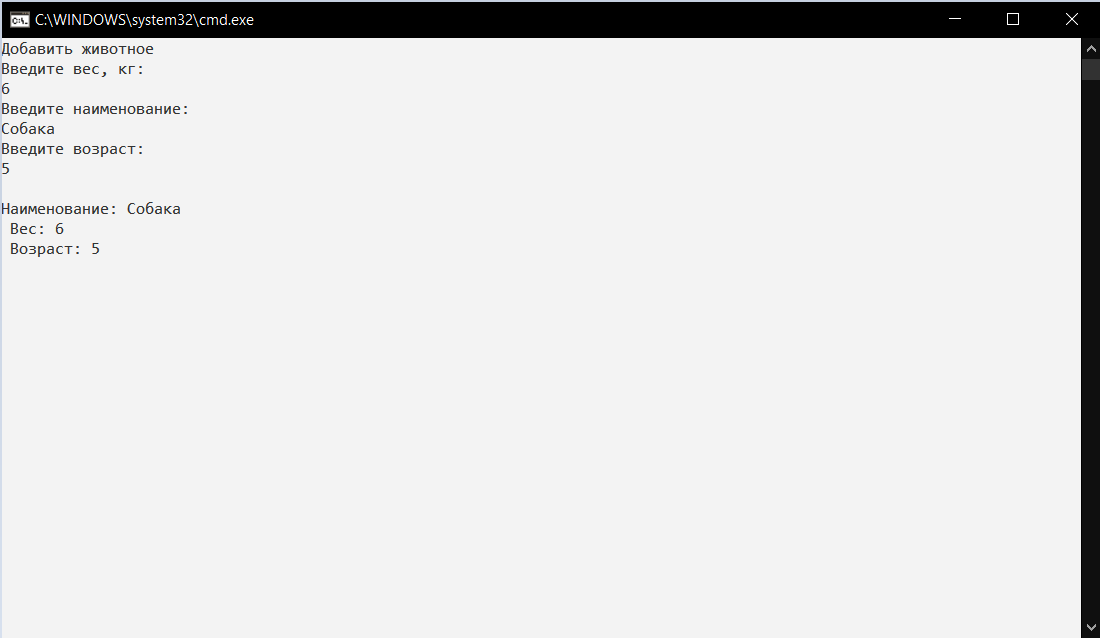


Рис. 2 – Заполнить данные о животном

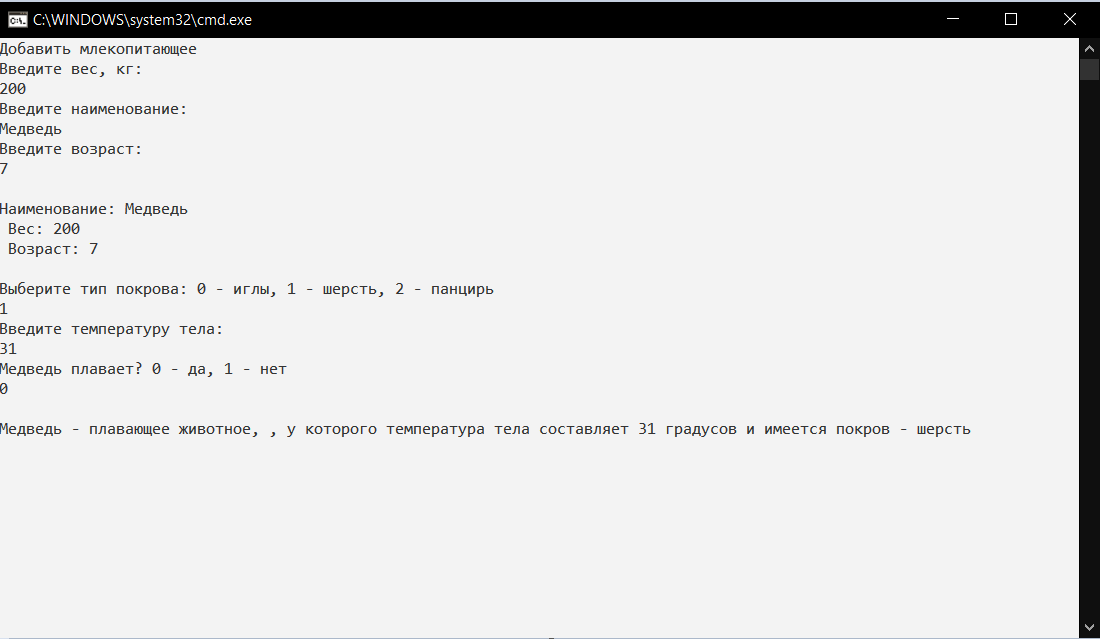


Рис. 3 – Заполнить данные о млекопитающем

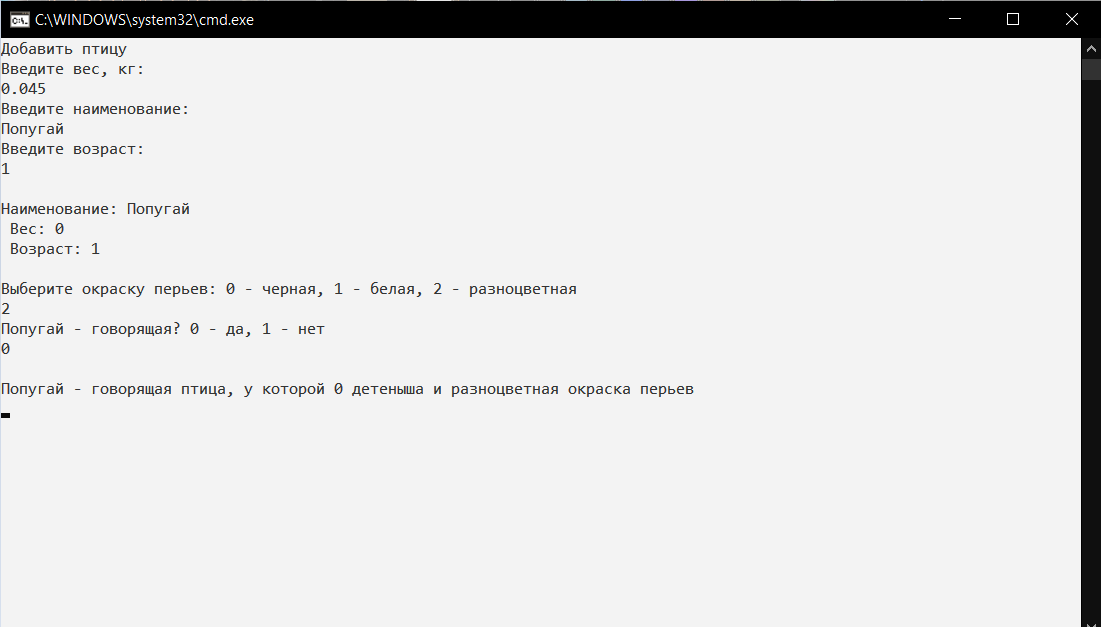


Рис. 4 – Заполнить данные о птице

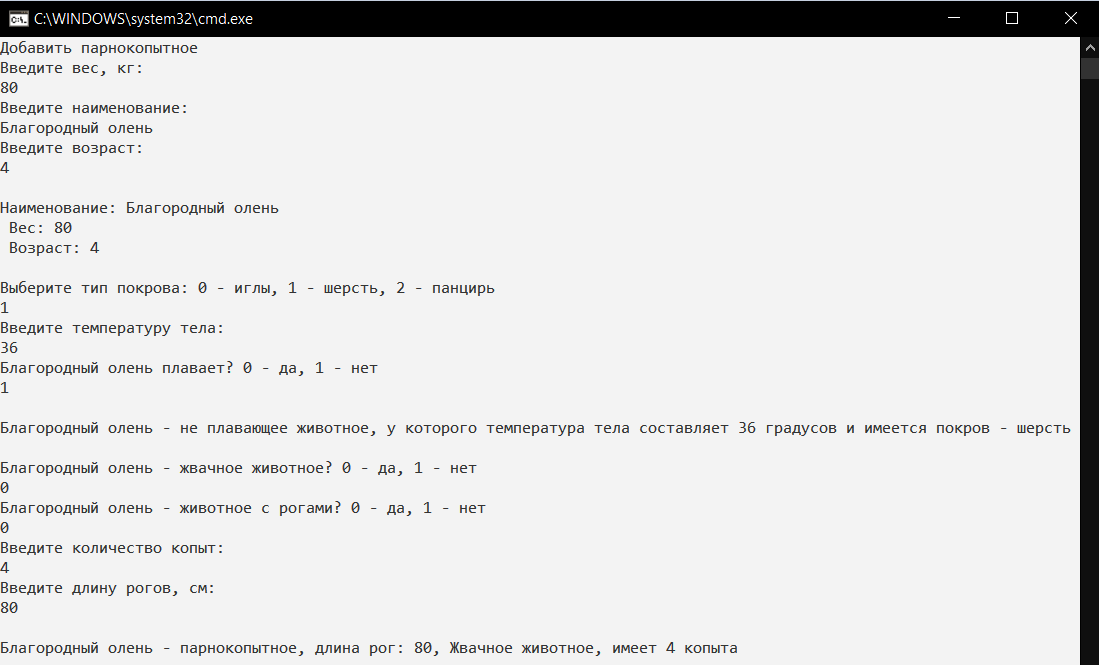


Рис. 5 – Заполнить данные о парнокопытном

*Листинг программных модулей:*

**Program.cs:**

using AnimalsEntity;

using System;

namespace AnimalsConsoleApp

{

class Program

{

static void Main(string[] args)

{

int age, HornLength, Eggs, choice, Hoofs = 0;

double Weight, Temperature = 0;

string Name, value;

bool IsRum, HornAvailable, IsTalking, IsSwimming, res;

Color Wings;

TypeCover Cover;

int menu;

for (; ;)

{

Console.WriteLine("1. Животное\n2. Млекопитающее\n3. Птица\n4. Парнокопытное\n0. Выход\n");

menu = int.Parse(Console.ReadLine());

switch (menu)

{

default:

{

Console.WriteLine("Выбран неверный пункт меню!");

break;

}

case 0:

{

return;

}

case 1:

{

Console.Clear();

Console.WriteLine("Добавить животное ");

Console.WriteLine("Введите вес, кг: ");

value = Console.ReadLine();

res = double.TryParse(value, out Weight);

Console.WriteLine("Введите наименование: ");

Name = Console.ReadLine();

Console.WriteLine("Введите возраст: ");

value = Console.ReadLine();

res = int.TryParse(value, out age);

Animal b = new Animal()

{

Age = age,

Name = Name,

Weight = Weight

};

Console.WriteLine("\n" + b.ToString());

Console.ReadLine();

Console.Clear();

break;

}

case 2:

{

Console.Clear();

Console.WriteLine("Добавить млекопитающее ");

Console.WriteLine("Введите вес, кг: ");

value = Console.ReadLine();

res = double.TryParse(value, out Weight);

Console.WriteLine("Введите наименование: ");

Name = Console.ReadLine();

Console.WriteLine("Введите возраст: ");

value = Console.ReadLine();

res = int.TryParse(value, out age);

Animal b = new Animal()

{

Age = age,

Name = Name,

Weight = Weight

};

Console.WriteLine("\n" + b.ToString());

Cover = TypeCover.Wool;

Console.WriteLine("Выберите тип покрова: 0 - иглы, 1 - шерсть, 2 - панцирь");

value = Console.ReadLine();

res = int.TryParse(value, out choice);

switch (choice)

{

case 0: Cover = TypeCover.Spines;

break;

case 1: Cover = TypeCover.Wool;

break;

case 2:

break;

default: Cover = TypeCover.Shell;

break;

}

Console.WriteLine("Введите температуру тела: ");

value = Console.ReadLine();

res = double.TryParse(value, out Temperature);

Console.WriteLine($"{Name} плавает? 0 - да, 1 - нет");

value = Console.ReadLine();

IsSwimming = (value == "0") ? true : false;

Mammal mam = new Mammal()

{

Age = b.Age,

Cover = Cover,

IsSwimming = IsSwimming,

Name = b.Name,

Temperature = Temperature,

Weight = Weight

};

Console.WriteLine("\n" + mam.ToString());

Console.ReadLine();

Console.Clear();

break;

}

case 3:

{

Console.Clear();

Console.WriteLine("Добавить птицу ");

Console.WriteLine("Введите вес, кг: ");

value = Console.ReadLine();

res = double.TryParse(value, out Weight);

Console.WriteLine("Введите наименование: ");

Name = Console.ReadLine();

Console.WriteLine("Введите возраст: ");

value = Console.ReadLine();

res = int.TryParse(value, out age);

Animal b = new Animal()

{

Age = age,

Name = Name,

Weight = Weight

};

Console.WriteLine("\n" + b.ToString());

Wings = Color.Black;

Console.WriteLine("Выберите окраску перьев: 0 - черная, 1 - белая, 2 - разноцветная");

value = Console.ReadLine();

res = int.TryParse(value, out choice);

switch (choice)

{

case 0:

Wings = Color.Black;

break;

case 1:

Wings = Color.White;

break;

case 2:

Wings = Color.Multicolored;

break;

default:

Wings = Color.Multicolored;

break;

}

Console.WriteLine($"{Name} - говорящая? 0 - да, 1 - нет");

value = Console.ReadLine();

IsTalking = (value == "0") ? true : false;

Bird mam = new Bird()

{

Age = b.Age,

IsTalking = IsTalking,

Name = b.Name,

Wings = Wings,

Weight = Weight

};

Console.WriteLine("\n" + mam.ToString());

Console.ReadLine();

Console.Clear();

break;

}

case 4:

{

Console.Clear();

HornLength = 0;

Console.WriteLine("Добавить парнокопытное ");

Console.WriteLine("Введите вес, кг: ");

value = Console.ReadLine();

res = double.TryParse(value, out Weight);

Console.WriteLine("Введите наименование: ");

Name = Console.ReadLine();

Console.WriteLine("Введите возраст: ");

value = Console.ReadLine();

res = int.TryParse(value, out age);

Animal b = new Animal()

{

Age = age,

Name = Name,

Weight = Weight

};

Console.WriteLine("\n" + b.ToString());

Cover = TypeCover.Wool;

Console.WriteLine("Выберите тип покрова: 0 - иглы, 1 - шерсть, 2 - панцирь");

value = Console.ReadLine();

res = int.TryParse(value, out choice);

switch (choice)

{

case 0:

Cover = TypeCover.Spines;

break;

case 1:

Cover = TypeCover.Wool;

break;

case 2:

Cover = TypeCover.Shell;

break;

default:

Cover = TypeCover.Shell;

break;

}

Console.WriteLine("Введите температуру тела: ");

value = Console.ReadLine();

res = double.TryParse(value, out Temperature);

Console.WriteLine($"{Name} плавает? 0 - да, 1 - нет");

value = Console.ReadLine();

IsSwimming = (value == "0") ? true : false;

Mammal mam = new Mammal()

{

Age = b.Age,

Cover = Cover,

IsSwimming = IsSwimming,

Name = b.Name,

Temperature = Temperature,

Weight = Weight

};

Console.WriteLine("\n" + mam.ToString());

Console.WriteLine("\n" + $"{Name} - жвачное животное? 0 - да, 1 - нет");

value = Console.ReadLine();

IsRum = (value == "0") ? true : false;

Console.WriteLine($"{Name} - животное с рогами? 0 - да, 1 - нет");

value = Console.ReadLine();

HornAvailable = (value == "0") ? true : false;

Console.WriteLine("Введите количество копыт: ");

value = Console.ReadLine();

res = int.TryParse(value, out Hoofs);

if (HornAvailable)

{

Console.WriteLine("Введите длину рогов, см: ");

value = Console.ReadLine();

res = int.TryParse(value, out HornLength);

}

Artiodactyls art = new Artiodactyls()

{

Age = b.Age,

Cover = Cover,

Name = b.Name,

Hoofs = Hoofs,

Weight = Weight,

HornLength = HornLength,

IsRum = IsRum,

HornAvailable = HornAvailable,

IsSwimming = IsSwimming,

Temperature = Temperature

};

Console.WriteLine("\n" + art.ToString());

Console.ReadLine();

Console.Clear();

break;

}

}

}

}

}

}

**Animal.cs:**

using System;

namespace AnimalsEntity

{

public class Animal

{

public Guid Id { get; set; } = Guid.NewGuid();

public double Weight { get; set; }

public string Name { get; set; }

public int Age { get; set; }

public string CompareWeight(double weight)

{

if (this.Weight < weight)

return "Этот зверь весит меньше";

else if (this.Weight == weight)

return "Этот зверь точно такого веса";

else return "Этот зверь тяжелее";

}

public override string ToString()

{

return $"Наименование: {Name}\n Вес: {Weight}\n Возраст: {Age}\n";

}

public string Sleep()

{

return "Животное заснуло.";

}

}

}

**Artiodactyls.cs :**

using System;

namespace AnimalsEntity

{

public class Artiodactyls: Mammal

{

public Guid Id { get; set; } = Guid.NewGuid();

//жвачное

public bool IsRum { get; set; }

//копыта

public int Hoofs { get; set; }

//рога

public bool HornAvailable { get; set; }

public int HornLength { get; set; }

public string ForceJump()

{

if (HornAvailable)

return "Зверь атакует";

else

return "Зверь убегает";

}

public string CompareHorn(int horn)

{

if (HornAvailable)

{

if (this.HornLength < horn)

return "У этого зверя длина рогов меньше";

else if (this.HornLength == horn)

return "Длина рогов одинаковые";

else return "У этого зверя длина рогов больше";

}

else return "Рогов нет";

}

public override string ToString()

{

string rum = (IsRum) ? "Жвачное" : "Не жвачное";

string horn = (HornAvailable) ? "длина рог: " + this.HornLength : "рогов нет ";

return $"{Name} - парнокопытное, {horn}, {rum} животное, имеет {Hoofs} копыта";

}

}

}

**Mammal.cs:**

using System;

namespace AnimalsEntity

{

public enum TypeCover

{

Spines, Wool, Shell

}

public class Mammal: Animal

{

public Guid Id { get; set; } = Guid.NewGuid();

public TypeCover Cover { get; set; }

public double Temperature { get; set; }

public bool IsSwimming { get; set; }

public string CompareTemperature(double temp)

{

if (this.Temperature < temp)

return "У этого зверя температура меньше";

else if (this.Temperature == temp)

return "У этого зверя температура такая же";

else return "У этого зверя температура больше";

}

public string Swim()

{

if (IsSwimming)

return "Животное плавает.";

else

return "Животное не умеет плавать.";

}

public string GiveMilk()

{

return "Молоко отдоенно.";

}

public override string ToString()

{

string rum = (IsSwimming) ? "плавающее животное, " : "не плавающее животное";

string wings = (Cover == TypeCover.Spines) ? "иглы" :

(Cover == TypeCover.Wool) ? "шерсть" : "панцирь";

return $"{Name} - {rum}, у которого температура тела составляет {Temperature} градусов и имеется покров - {wings}";

}

}

}

**Bird.cs :**

using System;

namespace AnimalsEntity

{

public enum Color

{

Black, White, Multicolored

}

public class Bird: Animal

{

public Bird()

{

this.Eggs = 0;

}

public Guid Id { get; set; } = Guid.NewGuid();

public bool IsTalking { get; set; }

public Color Wings { get; set; }

public int Eggs { get; set; }

public override string ToString()

{

string rum = (IsTalking) ? "говорящая птица" : "Не говорящая птица";

string wings = (Wings == Color.Black) ? "черная" :

(Wings == Color.White) ? "белая" : "разноцветная";

return $"{Name} - {rum}, у которой {Eggs} детеныша и {wings} окраска перьев";

}

public string MakeSong()

{

return "Чирик, чирик..";

}

public void MakeChick()

{

Eggs++;

}

}

}

# **Лабораторная работа №2**

Вариант 13

*Задание*: Реализовать оконное приложение (Windows Presentation Foundation).

Оконное приложение должно позволять создавать объекты, отображать список созданных объектов в табличной форме, выполнять поиск. Реализовать сериализацию и десериализацию классов в приложении. Интерфейс приложения должен быть дополнен кнопками для сохранения и загрузки классов из XML файлов.

Операции сохранения и загрузки XML файлов должны быть выполнены с использованием методов async и await.

*Результат работы программы:*

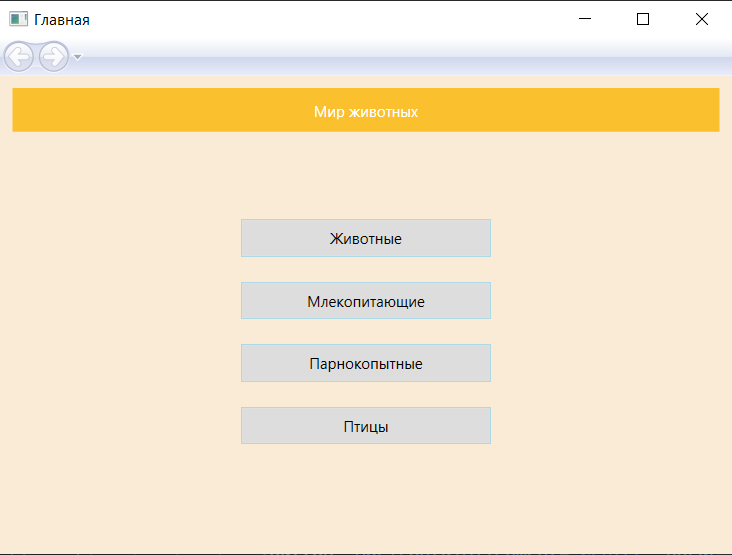


Рис. 6 – Главная страница

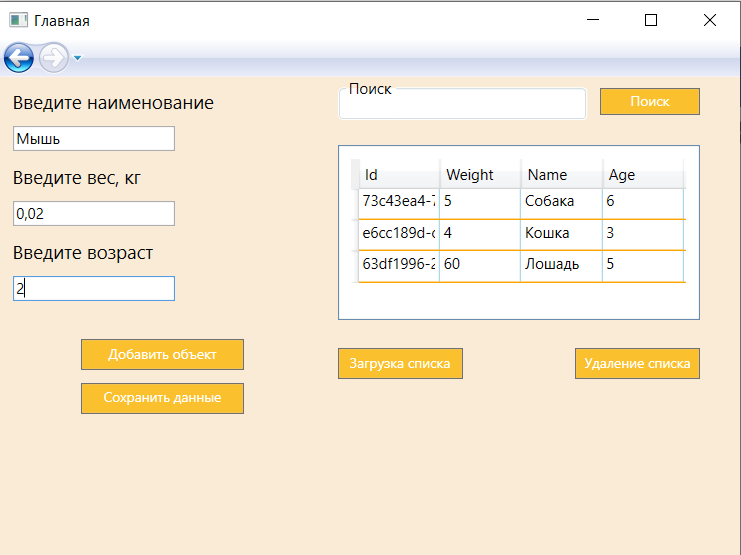


Рис. 7 – Форма заполнения сведений о животных

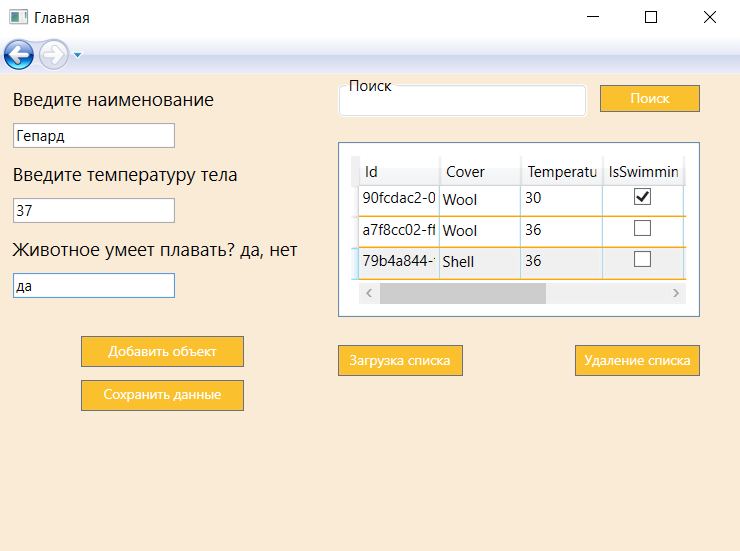


Рис. 8 – Форма заполнения сведений о млекопитающих

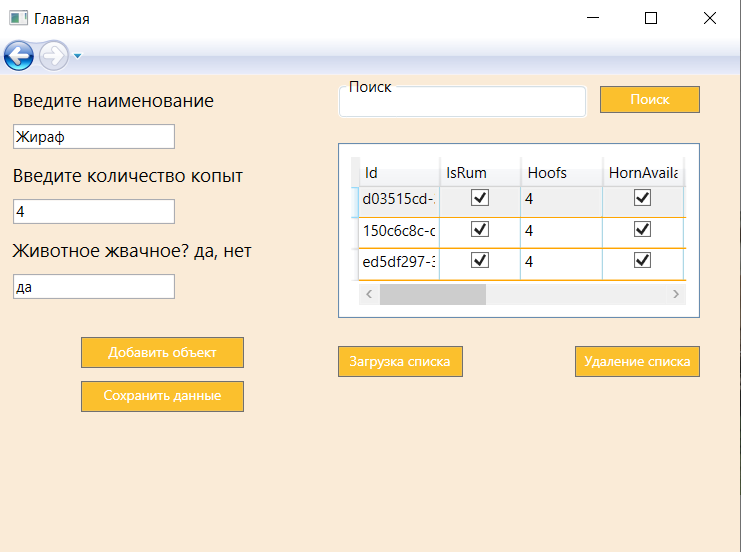


Рис. 9 – Форма заполнения сведений о парнокопытных

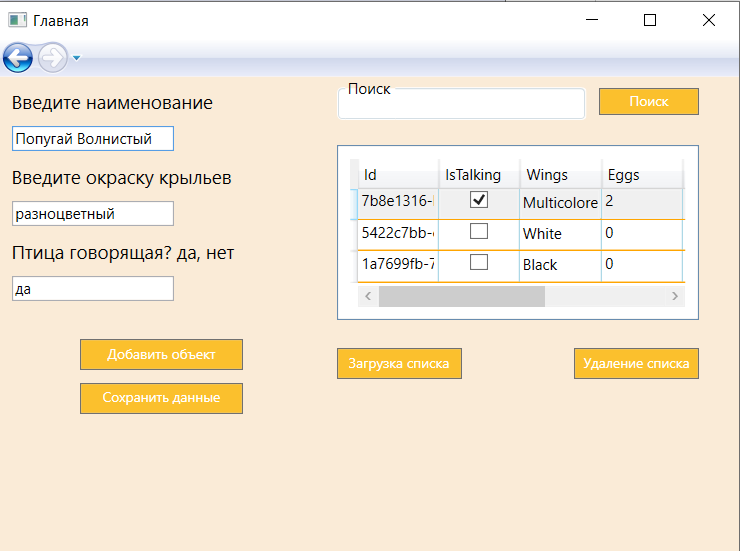


Рис. 10 – Форма заполнения сведений о птицах

*Листинг программных модулей:*

\*AnimalsEntity - тот же, что и в предыдущей лабораторной работе.

**HomePage.xaml:**

<Page x:Class="WpfAnimalsProject.HomePage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:WpfAnimalsProject"

mc:Ignorable="d"

Background="AntiqueWhite"

d:DesignHeight="380" d:DesignWidth="530"

Title="Главная">

<Grid>

<Grid Margin="10,10,10,10">

<StackPanel Orientation="Vertical">

<Border Height="35" Padding="5" Background="#FBC02D">

<Label HorizontalAlignment="Center" VerticalAlignment="Center" Foreground="White">Мир животных</Label>

</Border>

<Button HorizontalAlignment="Center" BorderBrush="LightBlue" FontSize="12" Width="200" Height="30" Margin="0,70,0,0" Click="Button\_Animal">Животные</Button>

<Button HorizontalAlignment="Center" BorderBrush="LightBlue" Height="30" Width="200" FontSize="12" Margin="0,20,0,0" Click="Button\_Mammal">Млекопитающие</Button>

<Button HorizontalAlignment="Center" BorderBrush="LightBlue" Height="30" Width="200" FontSize="12" Margin="0,20,0,0" Click="Button\_Anti">Парнокопытные</Button>

<Button HorizontalAlignment="Center" BorderBrush="LightBlue" Height="30" Width="200" FontSize="12" Margin="0,20,0,0" Click="Button\_Bird">Птицы</Button>

</StackPanel>

</Grid>

</Grid>

</Page>

**HomePage.xaml.cs :**

using System.Windows;

using System.Windows.Controls;

namespace WpfAnimalsProject

{

/// <summary>

/// Логика взаимодействия для HomePage.xaml

/// </summary>

public partial class HomePage : Page

{

public enum AnimalType

{

Animal, Mammal, Anti, Bird

}

public HomePage()

{

InitializeComponent();

}

private void Button\_Animal(object sender, RoutedEventArgs e)

{

// View Page

StaticAnimal.TypeOfAnimal = AnimalType.Animal;

AnimalPage expenseReportPage = new AnimalPage();

this.NavigationService.Navigate(expenseReportPage);

}

private void Button\_Mammal(object sender, RoutedEventArgs e)

{

// View Page

StaticAnimal.TypeOfAnimal = AnimalType.Mammal;

AnimalPage expenseReportPage = new AnimalPage();

this.NavigationService.Navigate(expenseReportPage);

}

private void Button\_Anti(object sender, RoutedEventArgs e)

{

// View Page

StaticAnimal.TypeOfAnimal = AnimalType.Anti;

AnimalPage expenseReportPage = new AnimalPage();

this.NavigationService.Navigate(expenseReportPage);

}

private void Button\_Bird(object sender, RoutedEventArgs e)

{

// View Page

StaticAnimal.TypeOfAnimal = AnimalType.Bird;

AnimalPage expenseReportPage = new AnimalPage();

this.NavigationService.Navigate(expenseReportPage);

}

}

}

**AnimalPage.xaml:**

<Page x:Class="WpfAnimalsProject.AnimalPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:WpfAnimalsProject"

mc:Ignorable="d"

Background="AntiqueWhite"

d:DesignHeight="300" d:DesignWidth="580"

Title="AnimalPage">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="260"/>

<ColumnDefinition Width="auto"/>

</Grid.ColumnDefinitions>

<StackPanel Orientation="Vertical" Margin="10" Grid.Column="0">

<TextBlock x:Name="txt\_first" FontSize="15" Text=""></TextBlock>

<TextBox Name="edit\_first" Width="130" HorizontalAlignment="Left" Height="20" Margin="0,10,0,0"></TextBox>

<TextBlock x:Name="txt\_second" Margin="0,10,0,0" FontSize="15" Text=""></TextBlock>

<TextBox Name="edit\_second" Width="130" HorizontalAlignment="Left" Height="20" Margin="0,10,0,0"></TextBox>

<TextBlock x:Name="txt\_third" Margin="0,10,0,0" FontSize="15" Text=""></TextBlock>

<TextBox Name="edit\_third" Width="130" HorizontalAlignment="Left" Height="20" Margin="0,10,0,0"></TextBox>

<Button Width="130" FontFamily="bold" FontWeight="Regular"

Padding="3" Height="25" Background="#FBC02D" Click="Button\_Add"

Foreground="White" Margin="0,30,0,0" FontSize="11">

Добавить объект

</Button>

<Button Width="130" FontFamily="bold" FontWeight="Regular"

Padding="3" Height="25" Background="#FBC02D" Click="Button\_Save"

Foreground="White" Margin="10" FontSize="11">

Сохранить данные

</Button>

</StackPanel>

<StackPanel Orientation="Vertical" Grid.Column="1">

<StackPanel Orientation="Horizontal">

<GroupBox Background="White" HorizontalAlignment="Left" Width="200" Header="Поиск" Margin="10,0,0,0" Height="35">

<TextBox Name="edit\_search" Foreground="Green" BorderBrush="White" Height="35" FontSize="10">

</TextBox>

</GroupBox>

<Button FontFamily="bold" FontWeight="Regular"

Margin="10,5,0,0" Width="80" Click="Button\_Search"

Padding="3" Height="22" Background="#FBC02D"

Foreground="White" FontSize="11">

Поиск

</Button>

</StackPanel>

<DataGrid BorderThickness="1"

Padding="10" Margin="10,20,0,0"

ColumnWidth="65" HeadersVisibility="All"

GridLinesVisibility="All" HorizontalGridLinesBrush="Orange"

VerticalGridLinesBrush="LightBlue" AutoGenerateColumns="true"

Height="140" VerticalAlignment="Top" Background="White"

AreRowDetailsFrozen="True" Width="290"

SelectionMode="Single" RowHeight="25" CanUserAddRows="False"

CanUserDeleteRows="False" ItemsSource="{Binding}"

Name="tb\_list" CanUserSortColumns = "False">

</DataGrid>

<StackPanel Orientation="Horizontal" Margin="0,17,0,0" HorizontalAlignment="Left">

<Button FontFamily="bold" FontWeight="Regular"

Margin="10,5,0,0" Width="100" Click="Button\_Load"

Padding="3" Height="25" Background="#FBC02D"

Foreground="White" FontSize="11">

Загрузка списка

</Button>

<Button FontFamily="bold" FontWeight="Regular"

Margin="90,5,0,0" Width="100" Click="Button\_Delete"

Padding="3" Height="25" Background="#FBC02D"

Foreground="White" FontSize="11">

Удаление списка

</Button>

</StackPanel>

</StackPanel>

</Grid>

</Page>

**AnimalPage.xaml.cs:**

using AnimalsEntity;

using System.Collections.Generic;

using System.ComponentModel;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Navigation;

using static WpfAnimalsProject.HomePage;

namespace WpfAnimalsProject

{

/// <summary>

/// Логика взаимодействия для AnimalPage.xaml

/// </summary>

public partial class AnimalPage : Page

{

AnimalType type;

public List<Animal> Animals = new List<Animal>();

public List<Artiodactyls> Artiodactyls = new List<Artiodactyls>();

public List<Mammal> Mammals = new List<Mammal>();

public List<Bird> Birds = new List<Bird>();

FileData<Animal> animal\_data;

FileData<Bird> bird\_data;

FileData<Artiodactyls> arti\_data;

FileData<Mammal> mammal\_data;

public AnimalPage()

{

InitializeComponent();

var parentWindow = this.Parent as Window;

NavigationService nav = NavigationService.GetNavigationService(this);

//if (parentWindow != null)

// parentWindow.Loaded += ParentWindow\_Loaded;

//this.NavigationService.LoadCompleted += NavigationService\_LoadCompleted;

type = StaticAnimal.TypeOfAnimal;

switch (type)

{

case AnimalType.Animal:

{

animal\_data = new FileData<Animal>(Animals, "animals.xml");

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите вес, кг";

txt\_third.Text = "Введите возраст";

}

break;

case AnimalType.Mammal:

{

mammal\_data = new FileData<Mammal>(Mammals, "mammals.xml");

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите температуру тела";

txt\_third.Text = "Животное умеет плавать? да, нет";

}

break;

case AnimalType.Anti:

{

arti\_data = new FileData<Artiodactyls>(Artiodactyls, "arti.xml");

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите количество копыт";

txt\_third.Text = "Животное жвачное? да, нет";

}

break;

case AnimalType.Bird:

{

bird\_data = new FileData<Bird>(Birds, "birds.xml");

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите окраску крыльев";

txt\_third.Text = "Птица говорящая? да, нет";

}

break;

default:

break;

}

}

private void Button\_Add(object sender, RoutedEventArgs e)

{

AddInfo();

}

private async void AddInfo()

{

if (string.IsNullOrWhiteSpace(txt\_first.Text) || string.IsNullOrWhiteSpace(txt\_second.Text)

|| string.IsNullOrWhiteSpace(txt\_third.Text))

{

MessageBox.Show("Введите все данные");

return;

}

switch (type)

{

case AnimalType.Animal:

{

Animals = await animal\_data.Deserialize();

Animal animal = new Animal();

animal.Age = int.Parse(edit\_third.Text);

animal.Name = edit\_first.Text;

animal.Weight = int.Parse(edit\_second.Text);

Animals.Add(animal);

animal\_data.objects = Animals;

animal\_data.XmlSerialize();

tb\_list.DataContext = Animals.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

case AnimalType.Mammal:

{

Mammals = await mammal\_data.Deserialize();

Mammal animal = new Mammal();

animal.Temperature = int.Parse(edit\_second.Text);

animal.Name = edit\_first.Text;

animal.IsSwimming = (edit\_third.Text == "да")? true: false;

Mammals.Add(animal);

mammal\_data.objects = Mammals;

mammal\_data.XmlSerialize();

tb\_list.DataContext = Mammals.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

case AnimalType.Anti:

{

Artiodactyls = await arti\_data.Deserialize();

Artiodactyls animal = new Artiodactyls();

animal.IsRum = (edit\_third.Text == "да") ? true : false;

animal.Name = edit\_first.Text;

animal.Hoofs= int.Parse(edit\_second.Text);

Artiodactyls.Add(animal);

arti\_data.objects = Artiodactyls;

arti\_data.XmlSerialize();

tb\_list.DataContext = Artiodactyls.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

case AnimalType.Bird:

{

Birds = await bird\_data.Deserialize();

Bird animal = new Bird();

animal.Wings = (edit\_second.Text == "черный") ? Color.Black : (edit\_second.Text == "белый") ? Color.White :

(edit\_second.Text == "разноцветный") ? Color.Multicolored : Color.Black;

animal.Name = edit\_first.Text;

animal.IsTalking = (edit\_third.Text == "да") ? true : false;

Birds.Add(animal);

bird\_data.objects = Birds;

bird\_data.XmlSerialize();

tb\_list.DataContext = Birds.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

default:

break;

}

}

private void Button\_Save(object sender, RoutedEventArgs e)

{

switch (type)

{

case AnimalType.Animal:

{

Animals = tb\_list.Items.OfType<Animal>().ToList();

animal\_data.objects = Animals;

animal\_data.XmlSerialize();

}

break;

case AnimalType.Mammal:

{

Mammals = tb\_list.Items.OfType<Mammal>().ToList();

mammal\_data.objects = Mammals;

mammal\_data.XmlSerialize();

}

break;

case AnimalType.Anti:

{

Artiodactyls = tb\_list.Items.OfType<Artiodactyls>().ToList();

arti\_data.objects = Artiodactyls;

arti\_data.XmlSerialize();

}

break;

case AnimalType.Bird:

{

Birds = tb\_list.Items.OfType<Bird>().ToList();

bird\_data.objects = Birds;

bird\_data.XmlSerialize();

}

break;

default:

break;

}

}

private void Button\_Load(object sender, RoutedEventArgs e)

{

LoadInfo();

}

private async void LoadInfo()

{

switch (type)

{

case AnimalType.Animal:

{

Animals = await animal\_data.Deserialize();

tb\_list.DataContext = Animals.ToList();

}

break;

case AnimalType.Mammal:

{

Mammals = await mammal\_data.Deserialize();

tb\_list.DataContext = Mammals.ToList();

}

break;

case AnimalType.Anti:

{

Artiodactyls = await arti\_data.Deserialize();

tb\_list.DataContext = Artiodactyls.ToList();

}

break;

case AnimalType.Bird:

{

Birds = await bird\_data.Deserialize();

tb\_list.DataContext = Birds.ToList();

}

break;

default:

break;

}

}

private void Button\_Search(object sender, RoutedEventArgs e)

{

switch (type)

{

case AnimalType.Animal:

{

BindingList<Animal> BS = new BindingList<Animal>(Animals.Where(m => m.Age.ToString().Contains(edit\_search.Text) || m.Name.ToLower().Contains(edit\_search.Text.ToLower()) || m.Weight.ToString().Contains(edit\_search.Text)).ToList());

tb\_list.DataContext = BS;

}

break;

case AnimalType.Mammal:

{

BindingList<Mammal> BS = new BindingList<Mammal>(Mammals.Where(m => m.Name.ToLower().Contains(edit\_search.Text.ToLower())).ToList());

tb\_list.DataContext = BS;

}

break;

case AnimalType.Anti:

{

BindingList<Artiodactyls> BS = new BindingList<Artiodactyls>(Artiodactyls.Where(m => m.Name.ToLower().Contains(edit\_search.Text.ToLower())).ToList());

tb\_list.DataContext = BS;

}

break;

case AnimalType.Bird:

{

BindingList<Bird> BS = new BindingList<Bird>(Birds.Where(m => m.Name.ToLower().Contains(edit\_search.Text.ToLower())).ToList());

tb\_list.DataContext = BS;

}

break;

default:

break;

}

}

private void Button\_Delete(object sender, RoutedEventArgs e)

{

switch (type)

{

case AnimalType.Animal:

{

Animals = new List<Animal>();

animal\_data.objects = Animals;

animal\_data.XmlSerialize();

tb\_list.DataContext = Animals.ToList();

}

break;

case AnimalType.Mammal:

{

Mammals = new List<Mammal>();

mammal\_data.objects = Mammals;

mammal\_data.XmlSerialize();

tb\_list.DataContext = Mammals.ToList();

}

break;

case AnimalType.Anti:

{

Artiodactyls = new List<Artiodactyls>();

arti\_data.objects = Artiodactyls;

arti\_data.XmlSerialize();

tb\_list.DataContext = Artiodactyls.ToList();

}

break;

case AnimalType.Bird:

{

Birds = new List<Bird>();

bird\_data.objects = Birds;

bird\_data.XmlSerialize();

tb\_list.DataContext = Birds.ToList();

}

break;

default:

break;

}

}

}

}

**FileData.cs :**

using System.Collections.Generic;

using System.IO;

using System.Threading.Tasks;

using System.Xml;

using System.Xml.Serialization;

namespace WpfAnimalsProject

{

public class FileData<T>

{

public List<T> objects { get; set; }

private string file\_name { get; set; }

public FileData(List<T> lst, string file\_name)

{

this.objects = new List<T>();

this.objects = lst;

this.file\_name = file\_name;

}

public async void XmlSerialize()

{

XmlWriter writer = new XmlTextWriter(file\_name, System.Text.Encoding.UTF8);

await Task.Run(() => {

XmlSerializer serializer = new XmlSerializer(typeof(List<T>));

serializer.Serialize(writer, objects);

});

writer.Close();

}

public async Task<List<T>> Deserialize()

{

var doc = new XmlDocument();

try

{

//doc.LoadXml(file\_name);

if (new FileInfo(file\_name).Length == 0)

{

return new List<T>();

}

List<T> LF = new List<T>();

XmlReader reader = new XmlTextReader(file\_name);

await Task.Run(() => {

XmlSerializer dserializer = new XmlSerializer(typeof(List<T>));

LF = (List<T>)dserializer.Deserialize(reader);

});

reader.Close();

return LF;

}

catch (XmlException e)

{

return new List<T>();

}

}

}

}

**StaticAnimal.cs :**

using static WpfAnimalsProject.HomePage;

namespace WpfAnimalsProject

{

public static class StaticAnimal

{

public static AnimalType TypeOfAnimal { get; set; }

}

}

# **Лабораторная работа №3**

Вариант 13

*Задание*: Создать веб-сервис, используя технологию веб-приложений ASP.NET и web-services (\*.asmx). Реализовать метод для запроса объектов с сервера (которые должны храниться в базе данных, для каждого класса, который задан в варианте). Реализовать методы для создания объектов каждого из классов заданного варианта на сервере (в метод передаются параметры, создание объекта происходит на сервере, объект записывается в базу данных). Клиентское приложение должно обращаться к веб-сервису, разработанному студентом с любым другим вариантом лабораторной работы.

Ссылка на веб-сервис должна быть добавлена в приложение, созданное в предыдущей лабораторной работе. Из оконного приложения запрос должен отправляться к веб-сервису для получения списка объектов и создания новых.

Запросы к серверу и их обработка должны использовать асинхронные варианты методов веб-сервиса (они генерируются автоматически для каждого соответствующего метода).

*Результат выполнения программы:*

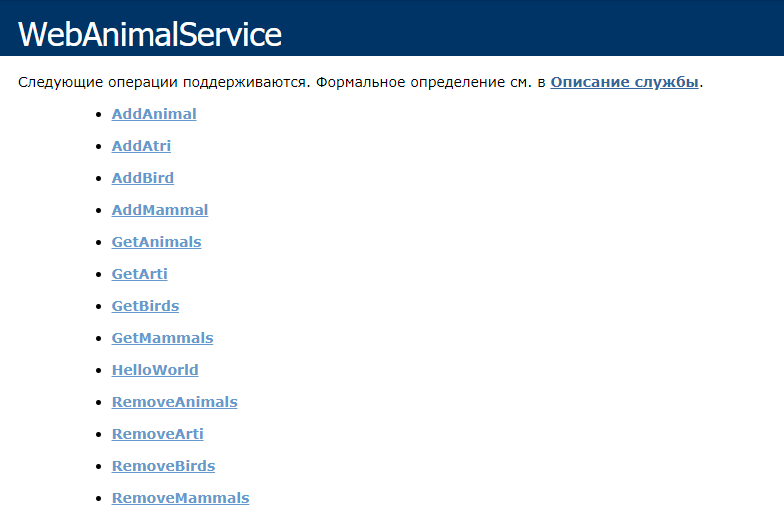


Рис. 11 – Общий вид веб-сервиса

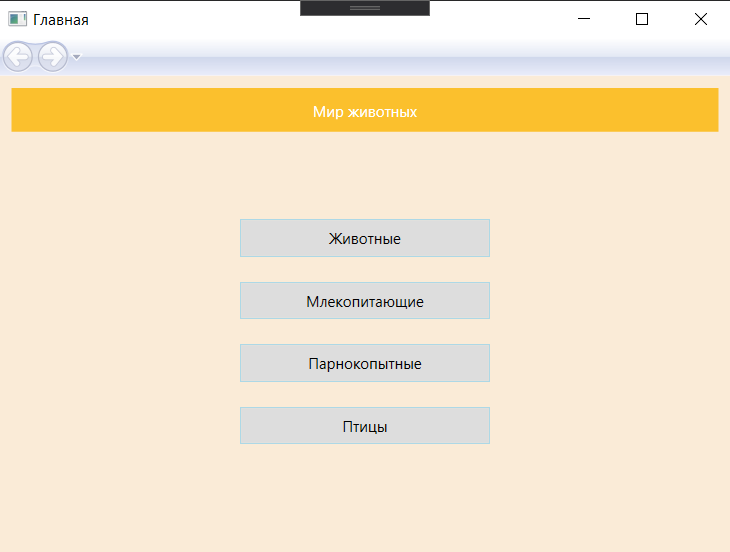


Рис. 12 – Главное меню

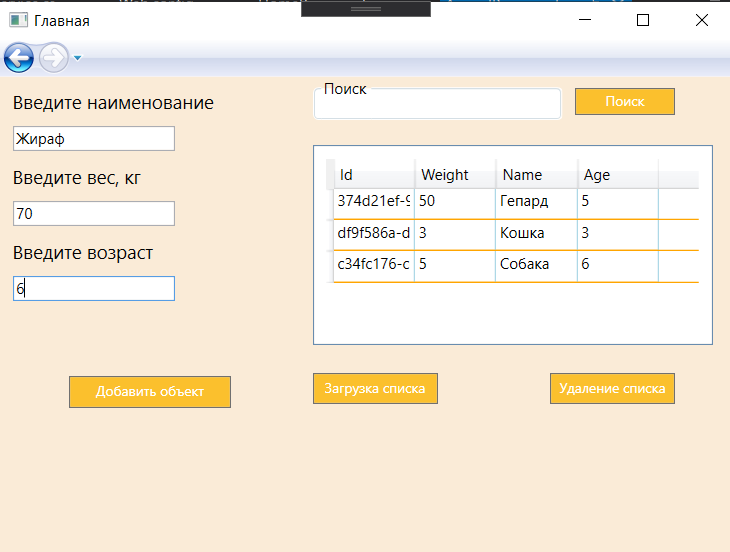


Рис. 13 – Оконное приложение, работающее с веб-сервисом

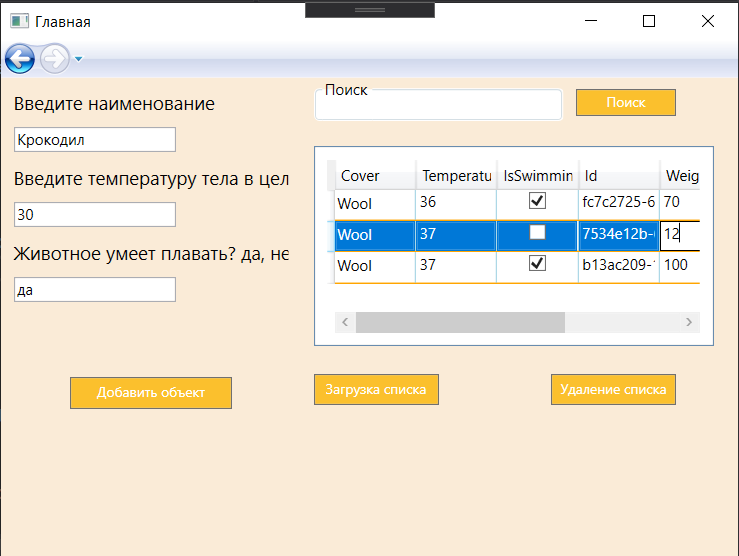


Рис. 14 – Оконное приложение, работающее с веб-сервисом

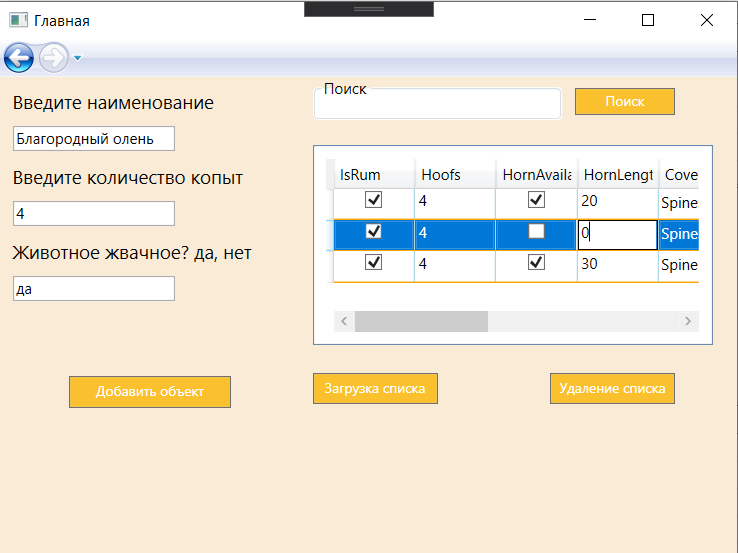


Рис. 15 – Оконное приложение, работающее с веб-сервисом

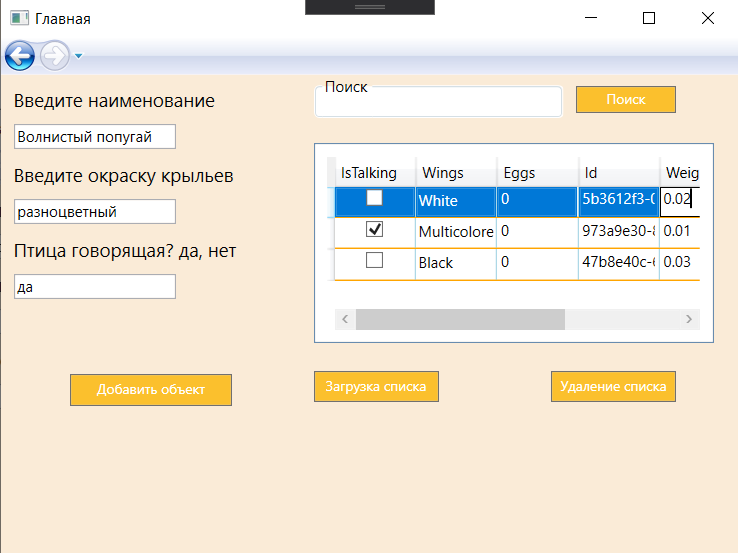


Рис. 16 – Оконное приложение, работающее с веб-сервисом

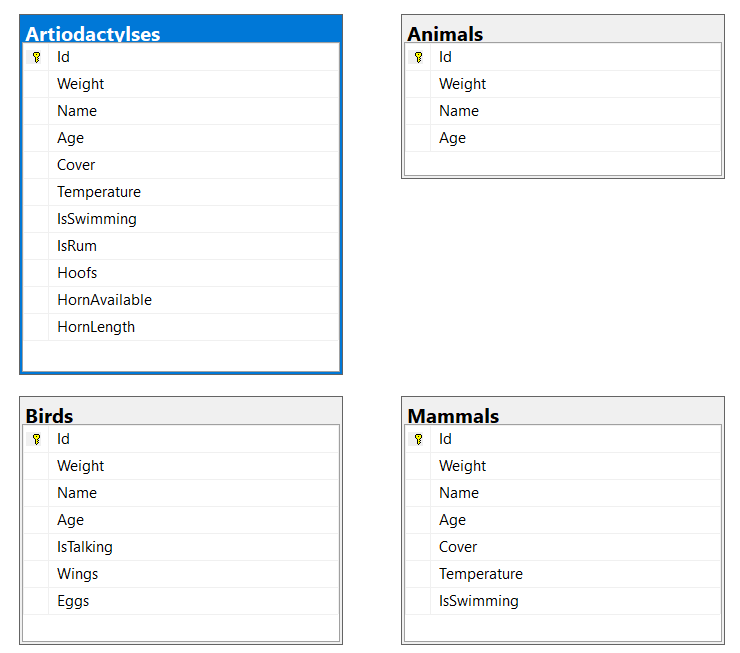


Рис. 17 – Модель базы данных

*Листинг программных модулей:*

**AnimalContext.cs:**

using AnimalsEntity;

using System.Data.Entity;

namespace WebAnimalProject.Context

{

public class AnimalContext: DbContext

{

public AnimalContext()

: base("DBConnection")

{

}

//protected override void OnModelCreating(DbModelBuilder modelBuilder)

//{

// throw new UnintentionalCodeFirstException();

//}

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

public virtual DbSet<Artiodactyls> Artiodactylses { get; set; }

public virtual DbSet<Mammal> Mammals { get; set; }

public virtual DbSet<Bird> Birds { get; set; }

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

modelBuilder.Entity<Animal>()

.Map(m =>

{

m.MapInheritedProperties();

m.ToTable("Animals");

});

modelBuilder.Entity<Artiodactyls>().Map(m =>

{

m.MapInheritedProperties();

m.ToTable("Artiodactylses");

});

modelBuilder.Entity<Mammal>().Map(m =>

{

m.MapInheritedProperties();

m.ToTable("Mammals");

});

modelBuilder.Entity<Bird>().Map(m =>

{

m.MapInheritedProperties();

m.ToTable("Birds");

});

}

}

}

**WebAnimalService.asmx.cs:**

using AnimalsEntity;

using System.Collections.Generic;

using System.Linq;

using System.Web.Services;

using WebAnimalProject.Context;

namespace WebAnimalProject.WebService

{

/// <summary>

/// Сводное описание для WebAnimalService

/// </summary>

[WebService(Namespace = "http://tempuri.org/")]

[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1\_1)]

[System.ComponentModel.ToolboxItem(false)]

// Чтобы разрешить вызывать веб-службу из скрипта с помощью ASP.NET AJAX, раскомментируйте следующую строку.

// [System.Web.Script.Services.ScriptService]

public class WebAnimalService : System.Web.Services.WebService

{

[WebMethod]

public string HelloWorld()

{

return "Привет всем!";

}

[WebMethod]

public bool AddAnimal(string name, int age, double weight)

{

using (AnimalContext bd = new AnimalContext())

{

Animal animal = new Animal

{

Age = age,

Name = name,

Weight = weight

};

bd.Animals.Add(animal);

bd.SaveChanges();

return true;

}

}

[WebMethod]

public bool AddMammal(string name, double temp, bool isIswimming)

{

using (AnimalContext bd = new AnimalContext())

{

Mammal animal = new Mammal

{

IsSwimming = isIswimming,

Name = name,

Temperature = temp

};

bd.Mammals.Add(animal);

bd.SaveChanges();

return true;

}

}

[WebMethod]

public bool AddAtri(string name, int h\_count, bool isRum)

{

using (AnimalContext bd = new AnimalContext())

{

Artiodactyls animal = new Artiodactyls

{

IsRum = isRum,

Name = name,

Hoofs = h\_count

};

bd.Artiodactylses.Add(animal);

bd.SaveChanges();

return true;

}

}

[WebMethod]

public bool AddBird(string name, Color wings, bool isTalking)

{

using (AnimalContext bd = new AnimalContext())

{

Bird animal = new Bird

{

Wings = wings,

Name = name,

IsTalking = isTalking

};

bd.Birds.Add(animal);

bd.SaveChanges();

return true;

}

}

[WebMethod]

public List<Animal> GetAnimals()

{

List<Animal> animals = new List<Animal>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Animals.ToList();

animals.AddRange(founded\_animal);

}

return animals;

}

[WebMethod]

public List<Mammal> GetMammals()

{

List<Mammal> animals = new List<Mammal>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Mammals.ToList();

animals.AddRange(founded\_animal);

}

return animals;

}

[WebMethod]

public List<Artiodactyls> GetArti()

{

List<Artiodactyls> animals = new List<Artiodactyls>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Artiodactylses.ToList();

animals.AddRange(founded\_animal);

}

return animals;

}

[WebMethod]

public List<Bird> GetBirds()

{

List<Bird> animals = new List<Bird>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Birds.ToList();

animals.AddRange(founded\_animal);

}

return animals;

}

[WebMethod]

public bool RemoveAnimals()

{

using (AnimalContext bd = new AnimalContext())

{

foreach (var item in bd.Animals)

{

bd.Animals.Remove(item);

}

bd.SaveChanges();

}

return true;

}

[WebMethod]

public bool RemoveMammals()

{

using (AnimalContext bd = new AnimalContext())

{

foreach (var item in bd.Mammals)

{

bd.Mammals.Remove(item);

}

bd.SaveChanges();

}

return true;

}

[WebMethod]

public bool RemoveArti()

{

using (AnimalContext bd = new AnimalContext())

{

foreach (var item in bd.Artiodactylses)

{

bd.Artiodactylses.Remove(item);

}

bd.SaveChanges();

}

return true;

}

[WebMethod]

public bool RemoveBirds()

{

using (AnimalContext bd = new AnimalContext())

{

foreach (var item in bd.Birds)

{

bd.Birds.Remove(item);

}

bd.SaveChanges();

}

return true;

}

}

}

**AnimalPage.xaml:**

<Page x:Class="WpfClientApp.AnimalPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:WpfClientApp"

mc:Ignorable="d"

Background="AntiqueWhite"

d:DesignHeight="300" d:DesignWidth="580"

Title="AnimalPage">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="240"/>

<ColumnDefinition Width="auto"/>

</Grid.ColumnDefinitions>

<StackPanel Orientation="Vertical" Margin="10" Grid.Column="0">

<TextBlock x:Name="txt\_first" FontSize="15" Text=""></TextBlock>

<TextBox Name="edit\_first" Width="130" HorizontalAlignment="Left" Height="20" Margin="0,10,0,0"></TextBox>

<TextBlock x:Name="txt\_second" Margin="0,10,0,0" FontSize="15" Text=""></TextBlock>

<TextBox Name="edit\_second" Width="130" HorizontalAlignment="Left" Height="20" Margin="0,10,0,0"></TextBox>

<TextBlock x:Name="txt\_third" Margin="0,10,0,0" FontSize="15" Text=""></TextBlock>

<TextBox Name="edit\_third" Width="130" HorizontalAlignment="Left" Height="20" Margin="0,10,0,0"></TextBox>

<Button Width="130" FontFamily="bold" FontWeight="Regular"

Padding="3" Height="25" Background="#FBC02D" Click="Button\_Add"

Foreground="White" Margin="0,60,0,0" FontSize="11">

Добавить объект

</Button>

</StackPanel>

<StackPanel Orientation="Vertical" Grid.Column="1">

<StackPanel Orientation="Horizontal">

<GroupBox Background="White" HorizontalAlignment="Left" Width="200" Header="Поиск" Margin="10,0,0,0" Height="35">

<TextBox Name="edit\_search" Foreground="Green" BorderBrush="White" Height="35" FontSize="10">

</TextBox>

</GroupBox>

<Button FontFamily="bold" FontWeight="Regular"

Margin="10,5,0,0" Width="80" Click="Button\_Search"

Padding="3" Height="22" Background="#FBC02D"

Foreground="White" FontSize="11">

Поиск

</Button>

</StackPanel>

<DataGrid BorderThickness="1"

Padding="10" Margin="10,20,0,0"

ColumnWidth="65" HeadersVisibility="All"

GridLinesVisibility="All" HorizontalGridLinesBrush="Orange"

VerticalGridLinesBrush="LightBlue" AutoGenerateColumns="true"

Height="160" VerticalAlignment="Top" Background="White"

AreRowDetailsFrozen="True" Width="320"

SelectionMode="Single" RowHeight="25" CanUserAddRows="False"

CanUserDeleteRows="False" ItemsSource="{Binding}"

Name="tb\_list" CanUserSortColumns = "False">

</DataGrid>

<StackPanel Orientation="Horizontal" Margin="0,17,0,0" HorizontalAlignment="Left">

<Button FontFamily="bold" FontWeight="Regular"

Margin="10,5,0,0" Width="100" Click="Button\_Load"

Padding="3" Height="25" Background="#FBC02D"

Foreground="White" FontSize="11">

Загрузка списка

</Button>

<Button FontFamily="bold" FontWeight="Regular"

Margin="90,5,0,0" Width="100" Click="Button\_Delete"

Padding="3" Height="25" Background="#FBC02D"

Foreground="White" FontSize="11">

Удаление списка

</Button>

</StackPanel>

</StackPanel>

</Grid>

</Page>

**AnimalPage.xaml.cs :**

using WpfClientApp.WebAnimalTestService;

using System;

using System.Collections.Generic;

using System.Linq;

using System.ServiceModel.Configuration;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using static WpfClientApp.HomePage;

using System.ComponentModel;

namespace WpfClientApp

{

/// <summary>

/// Логика взаимодействия для AnimalPage.xaml

/// </summary>

public partial class AnimalPage : Page

{

AnimalType type;

WebAnimalTestService.WebAnimalService service = new WebAnimalTestService.WebAnimalService();

public List<Animal> Animals = new List<Animal>();

public List<Artiodactyls> Artiodactyls = new List<Artiodactyls>();

public List<Mammal> Mammals = new List<Mammal>();

public List<Bird> Birds = new List<Bird>();

public AnimalPage()

{

InitializeComponent();

var parentWindow = this.Parent as Window;

NavigationService nav = NavigationService.GetNavigationService(this);

service.AddAnimalCompleted += Service\_AddAnimalCompleted;

service.AddAtriCompleted += Service\_AddAtriCompleted;

service.AddBirdCompleted += Service\_AddBirdCompleted;

service.AddMammalCompleted += Service\_AddMammalCompleted;

service.GetAnimalsCompleted += Service\_GetAnimalsCompleted;

service.GetArtiCompleted += Service\_GetArtiCompleted;

service.GetBirdsCompleted += Service\_GetBirdsCompleted;

service.GetMammalsCompleted += Service\_GetMammalsCompleted;

service.RemoveAnimalsCompleted += Service\_RemoveAnimalsCompleted;

service.RemoveArtiCompleted += Service\_RemoveArtiCompleted;

service.RemoveBirdsCompleted += Service\_RemoveBirdsCompleted;

service.RemoveMammalsCompleted += Service\_RemoveMammalsCompleted;

//if (parentWindow != null)

// parentWindow.Loaded += ParentWindow\_Loaded;

//this.NavigationService.LoadCompleted += NavigationService\_LoadCompleted;

type = StaticAnimal.TypeOfAnimal;

switch (type)

{

case AnimalType.Animal:

{

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите вес, кг";

txt\_third.Text = "Введите возраст";

}

break;

case AnimalType.Mammal:

{

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите температуру тела в цельсиях";

txt\_third.Text = "Животное умеет плавать? да, нет";

}

break;

case AnimalType.Anti:

{

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите количество копыт";

txt\_third.Text = "Животное жвачное? да, нет";

}

break;

case AnimalType.Bird:

{

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите окраску крыльев";

txt\_third.Text = "Птица говорящая? да, нет";

}

break;

default:

break;

}

}

private void Service\_RemoveMammalsCompleted(object sender, WebAnimalTestService.RemoveMammalsCompletedEventArgs e)

{

if (e.Result == true)

MessageBox.Show("Список млекопитающих успешно очищен!");

else

MessageBox.Show("Ошибка удаления");

}

private void Service\_RemoveBirdsCompleted(object sender, WebAnimalTestService.RemoveBirdsCompletedEventArgs e)

{

if (e.Result == true)

MessageBox.Show("Список птиц успешно очищен!");

else

MessageBox.Show("Ошибка удаления");

}

private void Service\_RemoveArtiCompleted(object sender, WebAnimalTestService.RemoveArtiCompletedEventArgs e)

{

if (e.Result == true)

MessageBox.Show("Список парнокопытных успешно очищен!");

else

MessageBox.Show("Ошибка удаления");

}

private void Service\_RemoveAnimalsCompleted(object sender, WebAnimalTestService.RemoveAnimalsCompletedEventArgs e)

{

if (e.Result == true)

MessageBox.Show("Список животных успешно очищен!");

else

MessageBox.Show("Ошибка удаления");

}

private void Service\_GetMammalsCompleted(object sender, WebAnimalTestService.GetMammalsCompletedEventArgs e)

{

Mammals = e.Result.ToList();

tb\_list.DataContext = Mammals;

}

private void Service\_GetBirdsCompleted(object sender, WebAnimalTestService.GetBirdsCompletedEventArgs e)

{

Birds = e.Result.ToList();

tb\_list.DataContext = Birds;

}

private void Service\_GetArtiCompleted(object sender, WebAnimalTestService.GetArtiCompletedEventArgs e)

{

Artiodactyls = e.Result.ToList();

tb\_list.DataContext = Artiodactyls;

}

private void Service\_GetAnimalsCompleted(object sender, WebAnimalTestService.GetAnimalsCompletedEventArgs e)

{

Animals = e.Result.ToList();

tb\_list.DataContext = Animals;

}

private void Service\_AddMammalCompleted(object sender, WebAnimalTestService.AddMammalCompletedEventArgs e)

{

if (e.Result == true)

MessageBox.Show("Млекопитающее успешно добавлено!");

else

MessageBox.Show("Ошибка добавления");

}

private void Service\_AddBirdCompleted(object sender, WebAnimalTestService.AddBirdCompletedEventArgs e)

{

if (e.Result == true)

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

else

MessageBox.Show("Ошибка добавления");

}

private void Service\_AddAtriCompleted(object sender, WebAnimalTestService.AddAtriCompletedEventArgs e)

{

if (e.Result == true)

MessageBox.Show("Парнокопытное успешно добавлено!");

else

MessageBox.Show("Ошибка добавления");

}

private void Service\_AddAnimalCompleted(object sender, WebAnimalTestService.AddAnimalCompletedEventArgs e)

{

if (e.Result == true)

MessageBox.Show("Животное успешно добавлено!");

else

MessageBox.Show("Ошибка добавления");

}

private void Button\_Add(object sender, RoutedEventArgs e)

{

AddInfo();

}

private void AddInfo()

{

if (string.IsNullOrWhiteSpace(txt\_first.Text) || string.IsNullOrWhiteSpace(txt\_second.Text)

|| string.IsNullOrWhiteSpace(txt\_third.Text))

{

MessageBox.Show("Введите все данные");

return;

}

switch (type)

{

case AnimalType.Animal:

{

Animal animal = new Animal();

animal.Age = int.Parse(edit\_third.Text);

animal.Name = edit\_first.Text;

animal.Weight = int.Parse(edit\_second.Text);

Animals.Add(animal);

service.AddAnimal(animal.Name, animal.Age, animal.Weight);

service.GetAnimalsAsync();

tb\_list.DataContext = Animals.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

case AnimalType.Mammal:

{

Mammal animal = new Mammal();

animal.Temperature = int.Parse(edit\_second.Text);

animal.Name = edit\_first.Text;

animal.IsSwimming = (edit\_third.Text == "да") ? true : false;

Mammals.Add(animal);

service.AddMammal(animal.Name, animal.Temperature, animal.IsSwimming);

service.GetMammalsAsync();

tb\_list.DataContext = Mammals.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

case AnimalType.Anti:

{

Artiodactyls animal = new Artiodactyls();

animal.IsRum = (edit\_third.Text == "да") ? true : false;

animal.Name = edit\_first.Text;

animal.Hoofs = int.Parse(edit\_second.Text);

Artiodactyls.Add(animal);

service.AddAtri(animal.Name, animal.Hoofs, animal.IsRum);

service.GetArtiAsync();

tb\_list.DataContext = Artiodactyls.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

case AnimalType.Bird:

{

Bird animal = new Bird();

animal.Wings = (edit\_second.Text == "черный") ? WebAnimalTestService.Color.Black : (edit\_second.Text == "белый") ? WebAnimalTestService.Color.White :

(edit\_second.Text == "разноцветный") ? WebAnimalTestService.Color.Multicolored : WebAnimalTestService.Color.Black;

animal.Name = edit\_first.Text;

animal.IsTalking = (edit\_third.Text == "да") ? true : false;

Birds.Add(animal);

service.AddBird(animal.Name, animal.Wings, animal.IsTalking);

service.GetBirdsAsync();

tb\_list.DataContext = Birds.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

default:

break;

}

}

private void Button\_Load(object sender, RoutedEventArgs e)

{

LoadInfo();

}

private void LoadInfo()

{

switch (type)

{

case AnimalType.Animal:

{

service.GetAnimalsAsync();

}

break;

case AnimalType.Mammal:

{

service.GetMammalsAsync();

}

break;

case AnimalType.Anti:

{

service.GetArtiAsync();

}

break;

case AnimalType.Bird:

{

service.GetBirdsAsync();

}

break;

default:

break;

}

}

private void Button\_Search(object sender, RoutedEventArgs e)

{

switch (type)

{

case AnimalType.Animal:

{

BindingList<Animal> BS = new BindingList<Animal>(Animals.Where(m => m.Age.ToString().Contains(edit\_search.Text) || m.Name.ToLower().Contains(edit\_search.Text.ToLower()) || m.Weight.ToString().Contains(edit\_search.Text)).ToList());

tb\_list.DataContext = BS;

}

break;

case AnimalType.Mammal:

{

BindingList<Mammal> BS = new BindingList<Mammal>(Mammals.Where(m => m.Name.ToLower().Contains(edit\_search.Text.ToLower())).ToList());

tb\_list.DataContext = BS;

}

break;

case AnimalType.Anti:

{

BindingList<Artiodactyls> BS = new BindingList<Artiodactyls>(Artiodactyls.Where(m => m.Name.ToLower().Contains(edit\_search.Text.ToLower())).ToList());

tb\_list.DataContext = BS;

}

break;

case AnimalType.Bird:

{

BindingList<Bird> BS = new BindingList<Bird>(Birds.Where(m => m.Name.ToLower().Contains(edit\_search.Text.ToLower())).ToList());

tb\_list.DataContext = BS;

}

break;

default:

break;

}

}

private void Button\_Delete(object sender, RoutedEventArgs e)

{

switch (type)

{

case AnimalType.Animal:

{

Animals = new List<Animal>();

service.RemoveAnimalsAsync();

tb\_list.DataContext = Animals.ToList();

}

break;

case AnimalType.Mammal:

{

Mammals = new List<Mammal>();

service.RemoveMammalsAsync();

tb\_list.DataContext = Mammals.ToList();

}

break;

case AnimalType.Anti:

{

Artiodactyls = new List<Artiodactyls>();

service.RemoveArtiAsync();

tb\_list.DataContext = Artiodactyls.ToList();

}

break;

case AnimalType.Bird:

{

Birds = new List<Bird>();

service.RemoveBirdsAsync();

tb\_list.DataContext = Birds.ToList();

}

break;

default:

break;

}

}

}

}

# **Лабораторная работа №4**

Вариант 13

*Задание*: К созданному веб-приложению необходимо добавить веб-страницу на основе технологии MVC с предсталвением\*.сshtml (страница ASP.NET). Затем необходимо в серверной части кода в контроллере (HttpPost) реализовать асинхронный метод и вызывать его с использованием технологий AJAX .

Реализованный на странице асинхронный метод должен возвращать объект одного из классов, после чего данный объект должен быть отображен на странице.

Варианты:

В соответствии с первой лабораторной работой.

При реализации лабораторной работы – во всплывающем сообщении должны отображаться значения полей какого-либо класса из варианта задания.

Изучить понятие JSON объектов, основы технологии JQuery.

*Результат работы программы:*

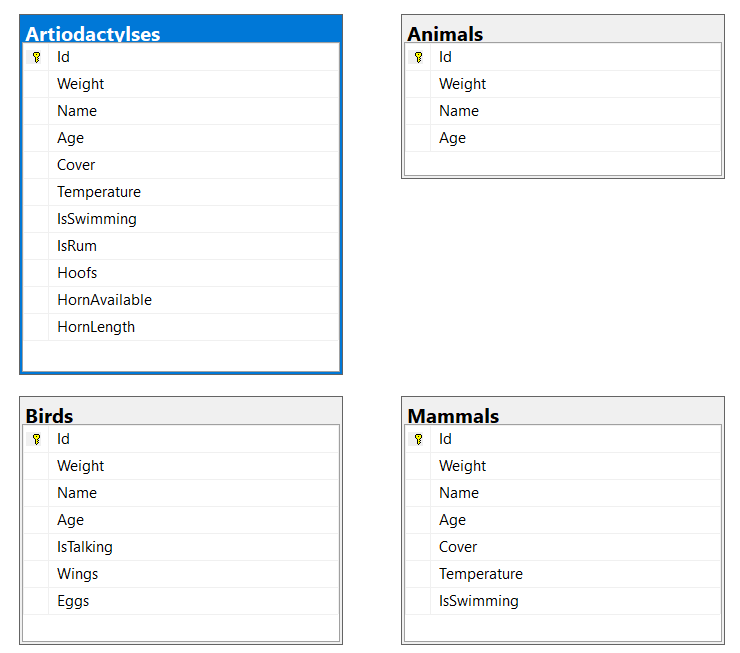


Рис. 18 – Модель базы данных

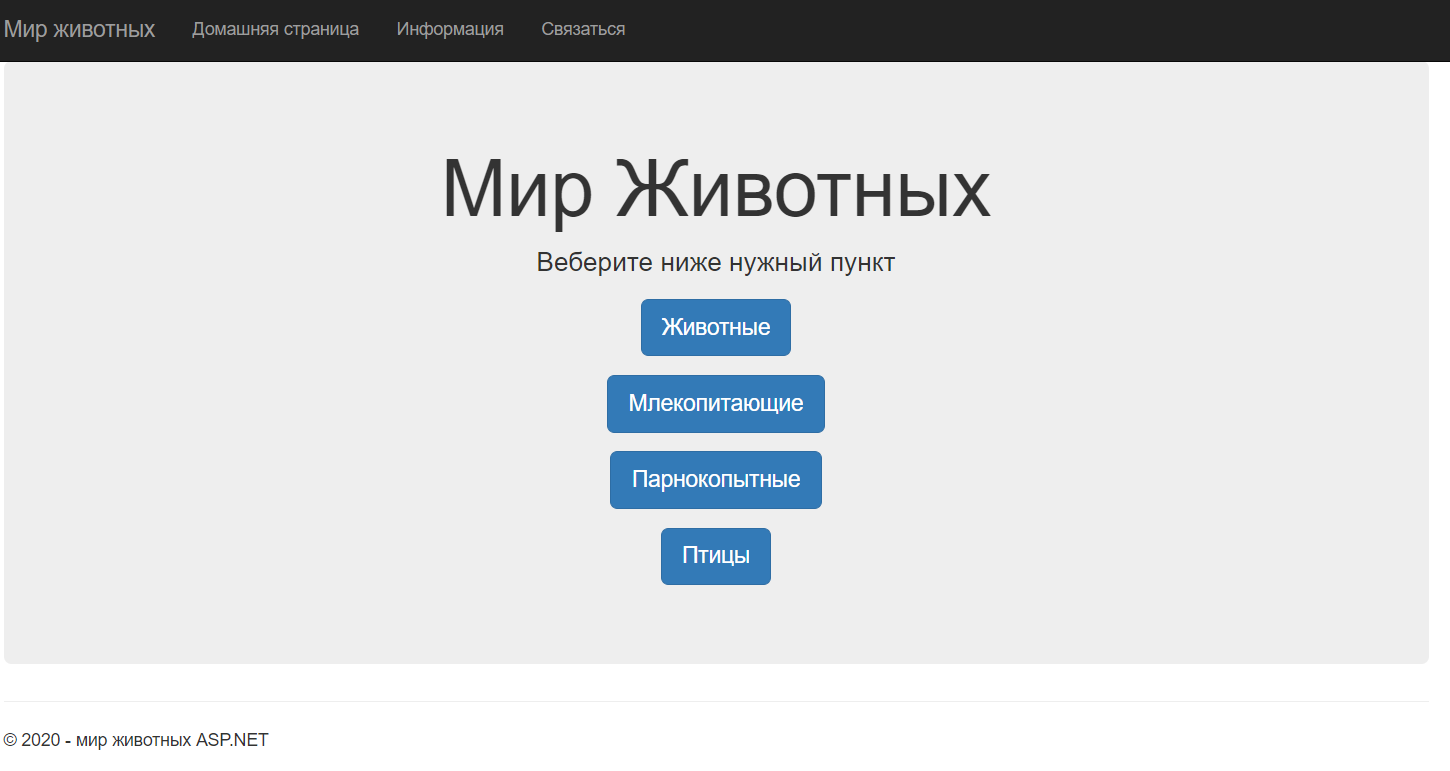


Рис. 19 – Главное меню

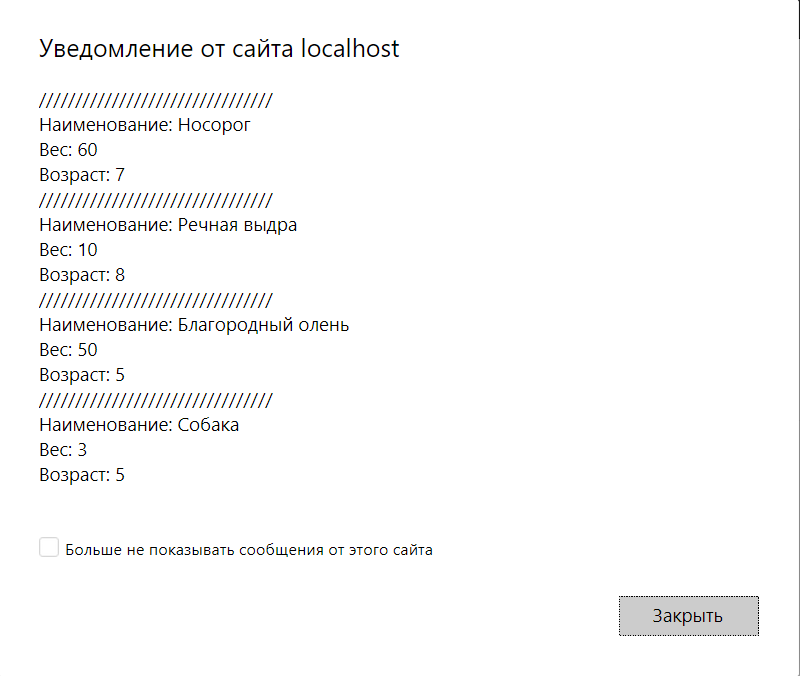


Рис. 20 – Отображение уведомления с перечнем добавленных животных

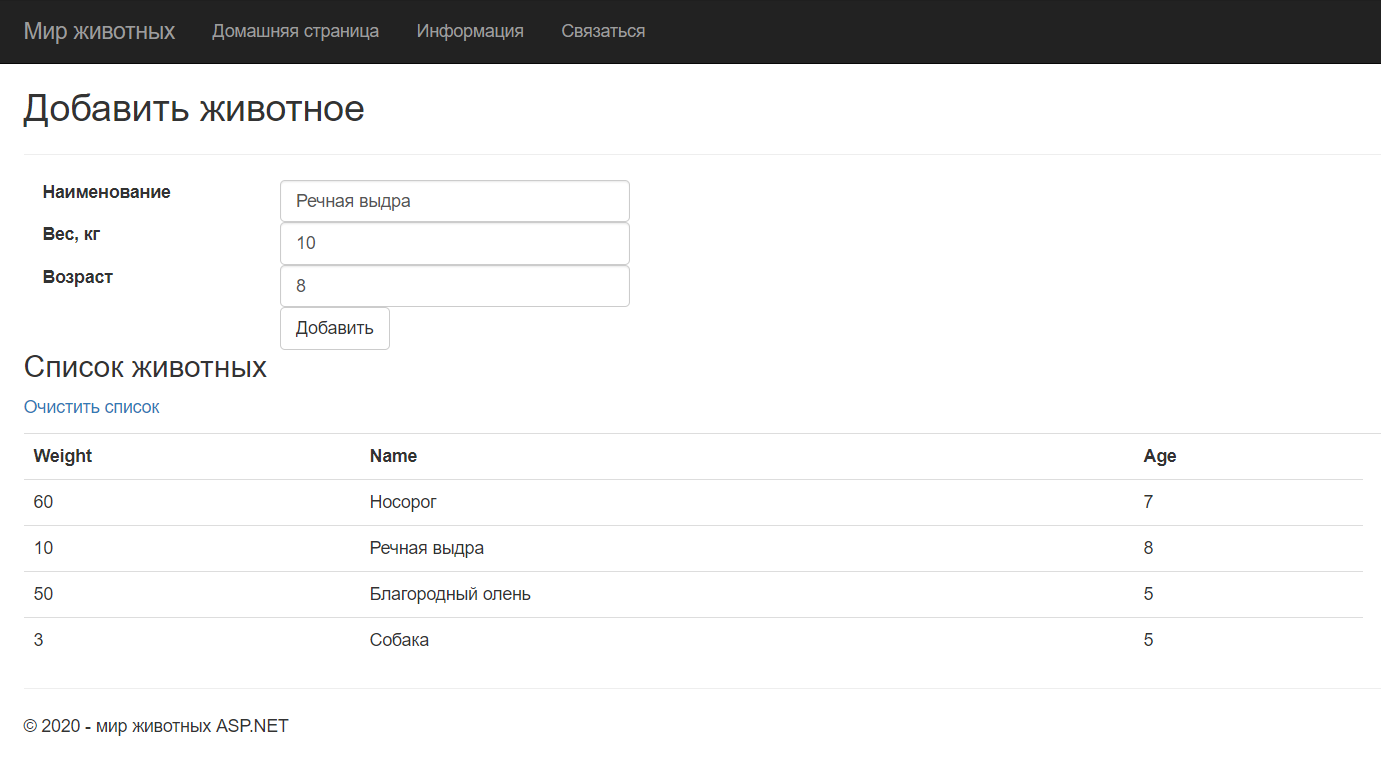


Рис.21 – Страница ASP.NET «Добавить животное»

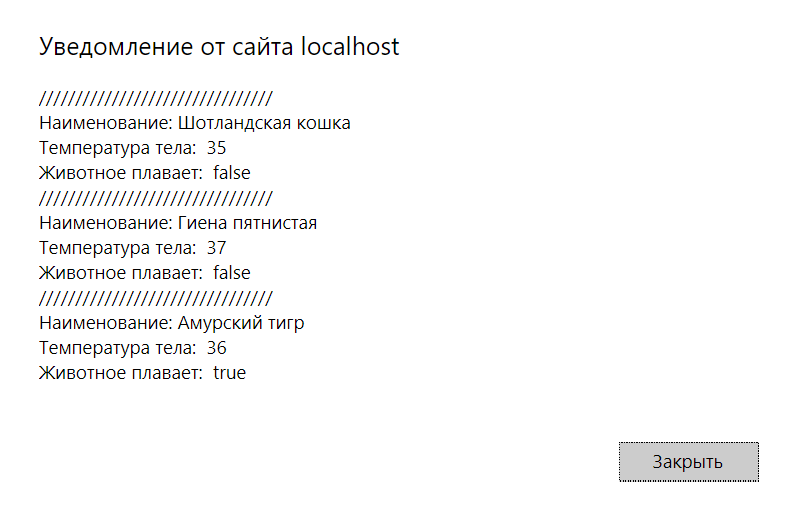


Рис. 22 – Отображение уведомления с перечнем добавленных млекопитающих

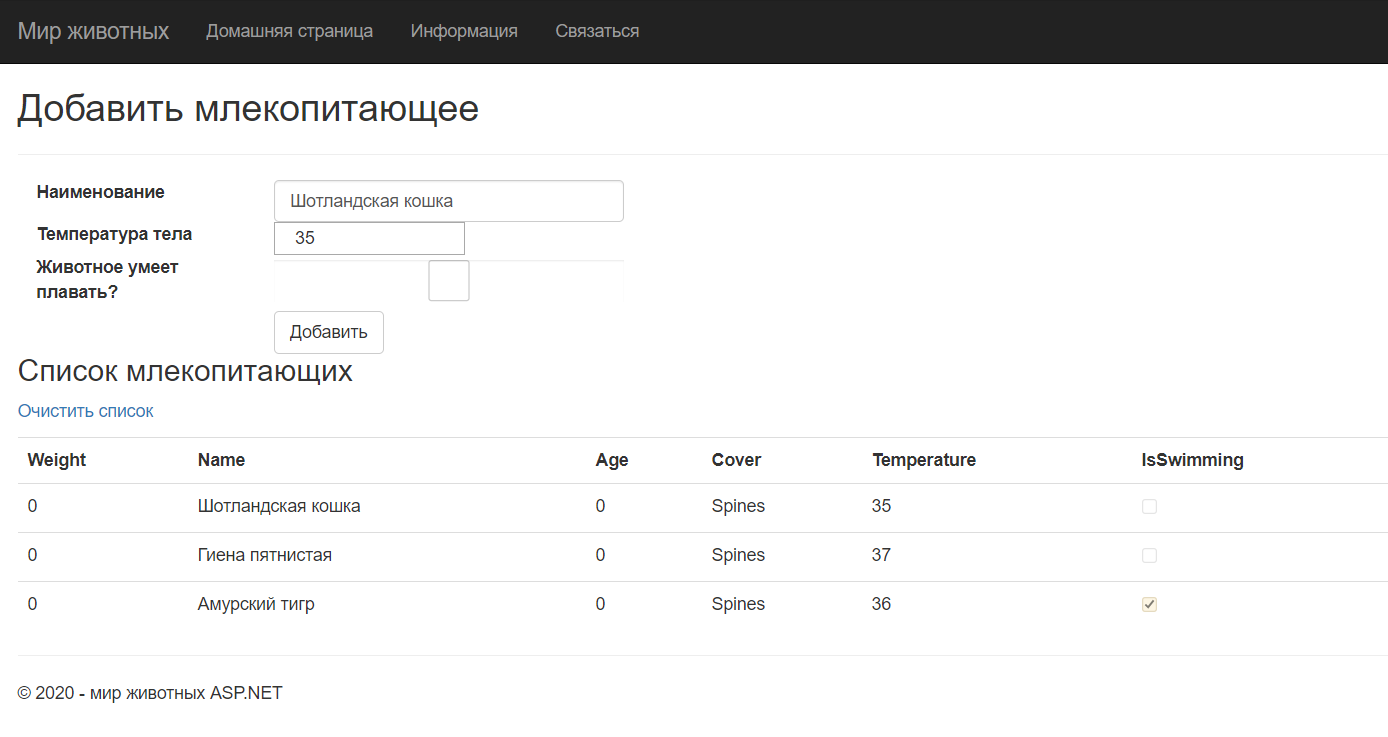


Рис.23 – Страница ASP.NET «Добавить млекопитающее»

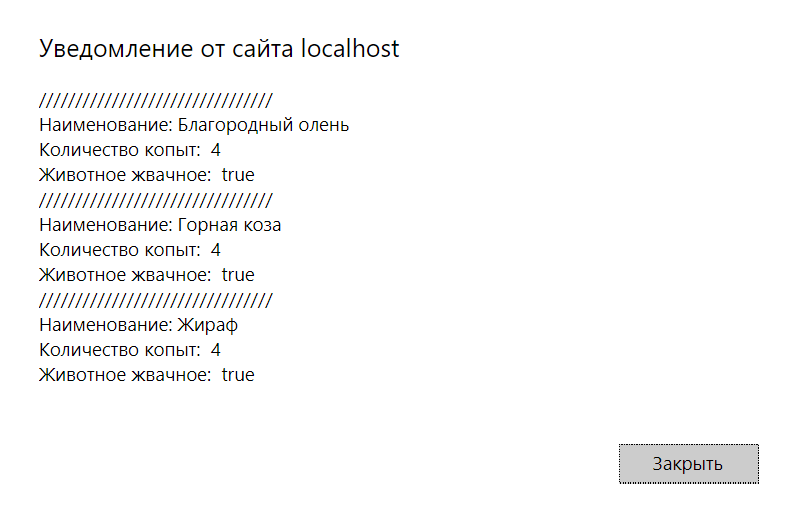


Рис. 24 – Отображение уведомления с перечнем добавленных парнокопытных

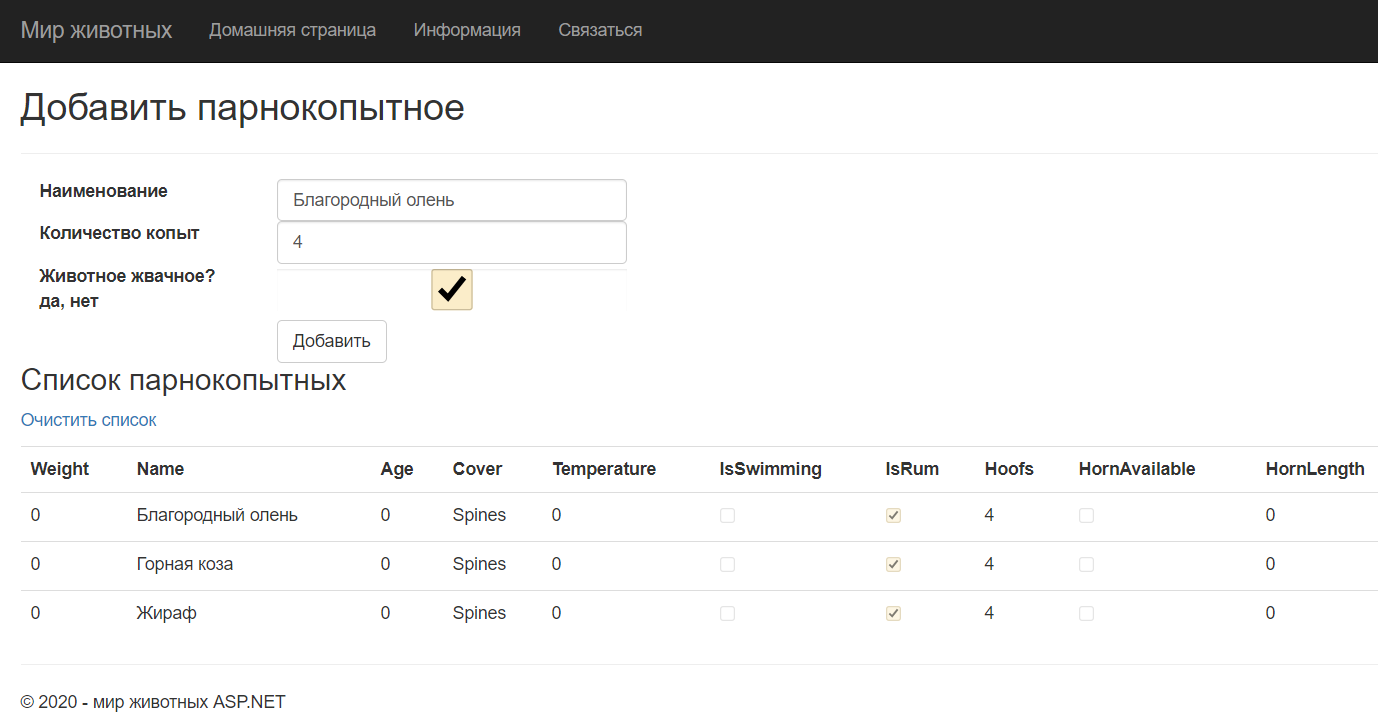


Рис.25 – Страница ASP.NET «Добавить парнокопытное»

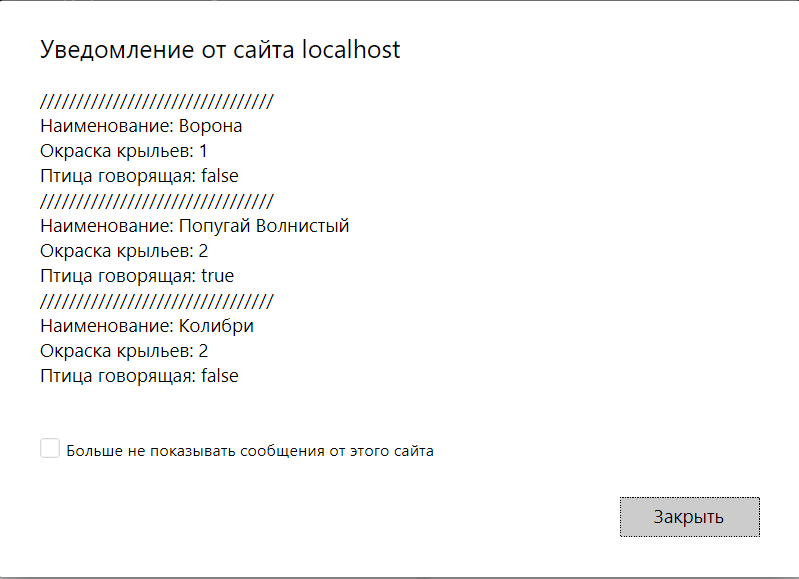


Рис. 26 – Отображение уведомления с перечнем добавленных птиц

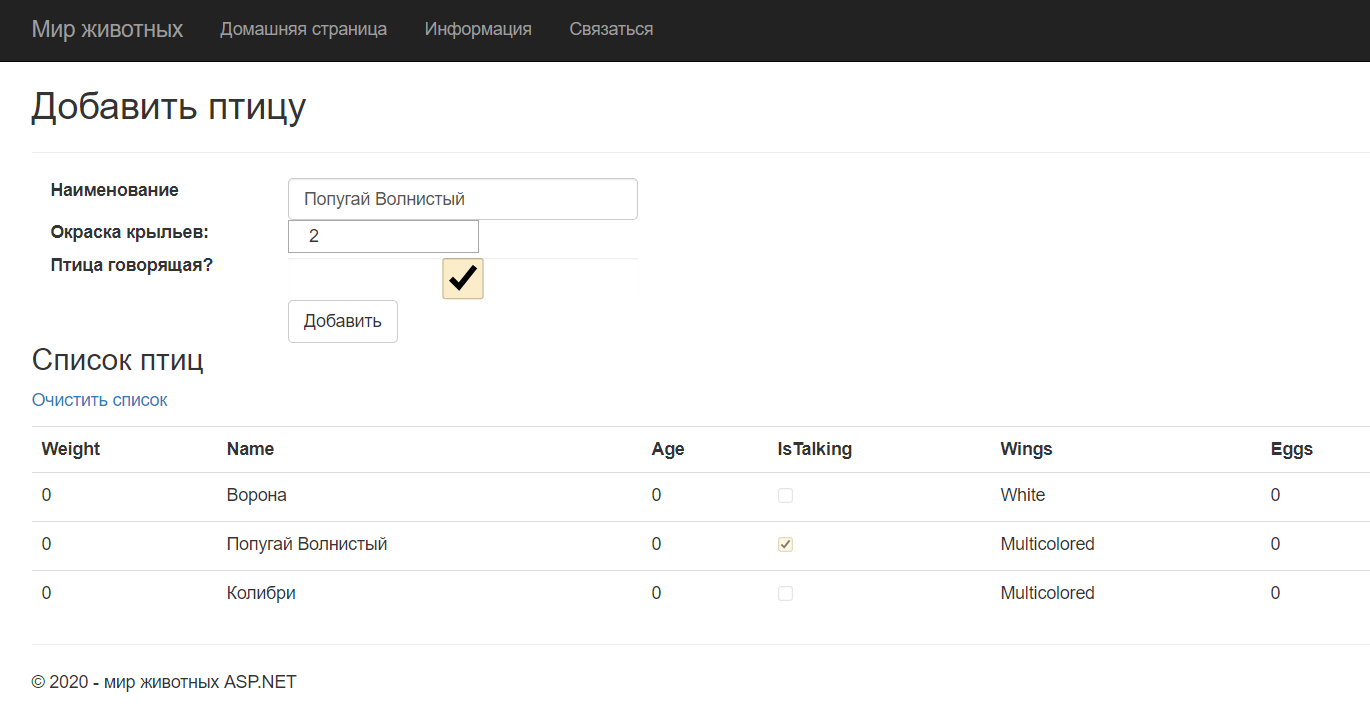


Рис.27 – Страница ASP.NET «Добавить птицу»

*Листинг программных модулей:*

**HomeController.cs:**

using AnimalsEntity;

using System.Collections.Generic;

using System.Linq;

using System.Web.Mvc;

using WebAnimalProject.Context;

namespace WebAnimalProject.Controllers

{

public class HomeController : Controller

{

public ActionResult Index()

{

return View();

}

public ActionResult About()

{

ViewBag.Message = "Your application description page.";

return View();

}

public ActionResult AnimalsView()

{

ViewBag.Message = "Список животных";

return View();

}

public ActionResult BirdsView()

{

SelectList list = new SelectList(new List<SelectListItem>

{

new SelectListItem { Selected = true, Text = "Черный", Value = "0"},

new SelectListItem { Selected = false, Text = "Разноцветный", Value = "1"},

new SelectListItem { Selected = false, Text = "Белый", Value = "2"},

});

ViewBag.Wings = list;

ViewBag.Message = "Список птиц";

return View();

}

public ActionResult MammalsView()

{

ViewBag.Message = "Список млекопитающих";

return View();

}

public ActionResult ArtiView()

{

ViewBag.Message = "Список парнокопытных";

return View();

}

[HttpPost]

public bool AddAnimal(Animal md)

{

using (AnimalContext bd = new AnimalContext())

{

Animal animal = new Animal

{

Age = md.Age,

Name = md.Name,

Weight = md.Weight

};

bd.Animals.Add(animal);

bd.SaveChanges();

return true;

}

}

[HttpPost]

public bool AddMammal(Mammal md)

{

using (AnimalContext bd = new AnimalContext())

{

Mammal animal = new Mammal

{

IsSwimming = md.IsSwimming,

Name = md.Name,

Temperature = md.Temperature

};

bd.Mammals.Add(animal);

bd.SaveChanges();

return true;

}

}

[HttpPost]

public bool AddArti(Artiodactyls art)

{

using (AnimalContext bd = new AnimalContext())

{

Artiodactyls animal = new Artiodactyls

{

IsRum = art.IsRum,

Name = art.Name,

Hoofs = art.Hoofs

};

bd.Artiodactylses.Add(animal);

bd.SaveChanges();

return true;

}

}

[HttpPost]

public bool AddBird(Bird bird)

{

using (AnimalContext bd = new AnimalContext())

{

Bird animal = new Bird

{

Wings = bird.Wings,

Name = bird.Name,

IsTalking = bird.IsTalking

};

bd.Birds.Add(animal);

bd.SaveChanges();

return true;

}

}

[HttpGet]

public ActionResult Animals()

{

List<Animal> animals = new List<Animal>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Animals.ToList();

animals.AddRange(founded\_animal);

}

return PartialView("Animals", animals);

}

[HttpGet]

public JsonResult AnimalsJson()

{

List<Animal> animals = new List<Animal>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Animals.ToList();

animals.AddRange(founded\_animal);

}

return Json(animals, JsonRequestBehavior.AllowGet);

}

[HttpGet]

public ActionResult Mammals()

{

List<Mammal> animals = new List<Mammal>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Mammals.ToList();

animals.AddRange(founded\_animal);

}

return PartialView(animals);

}

[HttpGet]

public JsonResult MammalsJson()

{

List<Mammal> animals = new List<Mammal>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Mammals.ToList();

animals.AddRange(founded\_animal);

}

return Json(animals, JsonRequestBehavior.AllowGet);

}

[HttpGet]

public ActionResult Arti()

{

List<Artiodactyls> animals = new List<Artiodactyls>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Artiodactylses.ToList();

animals.AddRange(founded\_animal);

}

return PartialView(animals);

}

[HttpGet]

public JsonResult ArtiJson()

{

List<Artiodactyls> animals = new List<Artiodactyls>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Artiodactylses.ToList();

animals.AddRange(founded\_animal);

}

return Json(animals, JsonRequestBehavior.AllowGet);

}

[HttpGet]

public ActionResult Birds()

{

List<Bird> animals = new List<Bird>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Birds.ToList();

animals.AddRange(founded\_animal);

}

return PartialView(animals);

}

[HttpGet]

public JsonResult BirdsJson()

{

List<Bird> animals = new List<Bird>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Birds.ToList();

animals.AddRange(founded\_animal);

}

return Json(animals, JsonRequestBehavior.AllowGet);

}

[HttpDelete]

public bool RemoveAnimals()

{

using (AnimalContext bd = new AnimalContext())

{

foreach (var item in bd.Animals)

{

bd.Animals.Remove(item);

}

bd.SaveChanges();

}

return true;

}

[HttpDelete]

public bool RemoveMammals()

{

using (AnimalContext bd = new AnimalContext())

{

foreach (var item in bd.Mammals)

{

bd.Mammals.Remove(item);

}

bd.SaveChanges();

}

return true;

}

[HttpDelete]

public bool RemoveArti()

{

using (AnimalContext bd = new AnimalContext())

{

foreach (var item in bd.Artiodactylses)

{

bd.Artiodactylses.Remove(item);

}

bd.SaveChanges();

}

return true;

}

[HttpDelete]

public bool RemoveBirds()

{

using (AnimalContext bd = new AnimalContext())

{

foreach (var item in bd.Birds)

{

bd.Birds.Remove(item);

}

bd.SaveChanges();

}

return true;

}

}

}

**animals.js:**

$(document).ready(function () {

});

function OnSuccessAnimals(data) {

$.ajax({

url: "https://localhost:44352/Home/AnimalsJson",

contentType: "application/json",

method: "GET",

cache: false,

success: function (response) {

var data = "";

for (var i = 0; i < response.length; i++) {

data += "////////////////////////////////" + "\n" + "Наименование: " +

response[i].Name + "\n" + "Вес: " + response[i].Weight + "\n" + "Возраст: " + response[i].Age + "\n";

}

alert(data);

},

error: function (jxqr, error, status) {

alert("Ошибка " + jxqr.statusText)

}

})

}

function OnSuccessAddAnimal(data) {

alert("Объект успешно добавлен");

}

function OnSuccessRemoveAnimals(data) {

alert("Список животных очищен!");

}

function OnSuccessBirds(data) {

$.ajax({

url: "https://localhost:44352/Home/BirdsJson",

contentType: "application/json",

method: "GET",

cache: false,

success: function (response) {

var data = "";

for (var i = 0; i < response.length; i++) {

data += "////////////////////////////////" + "\n" + "Наименование: " +

response[i].Name + "\n" + "Окраска крыльев: " + response[i].Wings + "\n" + "Птица говорящая: " + response[i].IsTalking + "\n";

}

alert(data);

},

error: function (jxqr, error, status) {

alert("Ошибка " + jxqr.statusText)

}

})

}

function OnSuccessAddBird(data) {

alert("Объект успешно добавлен");

}

function OnSuccessRemoveBirds(data) {

alert("Список птиц очищен!");

}

function OnSuccessMammals(data) {

$.ajax({

url: "https://localhost:44352/Home/MammalsJson",

contentType: "application/json",

method: "GET",

cache: false,

success: function (response) {

var data = "";

for (var i = 0; i < response.length; i++) {

data += "////////////////////////////////" + "\n" + "Наименование: " +

response[i].Name + "\n" + "Температура тела: " + response[i].Temperature + "\n" + "Животное плавает: " + response[i].IsSwimming + "\n";

}

alert(data);

},

error: function (jxqr, error, status) {

alert("Ошибка " + jxqr.statusText)

}

})

}

function OnSuccessAddMammal(data) {

alert("Объект успешно добавлен");

}

function OnSuccessRemoveMammals(data) {

alert("Список млекопитающих очищен!");

}

function OnSuccessArti(data) {

$.ajax({

url: "https://localhost:44352/Home/ArtiJson",

contentType: "application/json",

method: "GET",

cache: false,

success: function (response) {

var data = "";

for (var i = 0; i < response.length; i++) {

data += "////////////////////////////////" + "\n" + "Наименование: " +

response[i].Name + "\n" + "Количество копыт: " + response[i].Hoofs + "\n" + "Животное жвачное: " + response[i].IsRum + "\n";

}

alert(data);

},

error: function (jxqr, error, status) {

alert("Ошибка " + jxqr.statusText)

}

})

}

function OnSuccessAddArti(data) {

alert("Объект успешно добавлен");

}

function OnSuccessRemoveArti(data) {

alert("Список парнокопытных очищен!");

}

**Animals.cshtml:**

@model IEnumerable<AnimalsEntity.Animal>

<div id="animals">

<p>

@Ajax.ActionLink("Очистить список", "RemoveAnimals", new AjaxOptions { OnSuccess = "OnSuccessRemoveAnimals", HttpMethod="Delete"})

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.Weight)

</th>

<th>

@Html.DisplayNameFor(model => model.Name)

</th>

<th>

@Html.DisplayNameFor(model => model.Age)

</th>

<th></th>

</tr>

@foreach (var item in Model)

{

<tr>

<td>

@Html.DisplayFor(modelItem => item.Weight)

</td>

<td>

@Html.DisplayFor(modelItem => item.Name)

</td>

<td>

@Html.DisplayFor(modelItem => item.Age)

</td>

</tr>

}

</table>

</div>

**AnimalsView.cshtml:**

@using AnimalsEntity;

@model AnimalsEntity.Animal

@{

ViewBag.Title = "Добавить животное";

}

<h2>@ViewBag.Title</h2>

<div>

@using (Ajax.BeginForm("AddAnimal", new AjaxOptions { OnSuccess = "OnSuccessAddAnimal", HttpMethod = "Post" }))

{

<hr />

<div class="form-group">

@Html.Label("Наименование", new { @class = "col-md-2 control-label" })

<div class="col-md-10">

@Html.TextBoxFor(m => m.Name, new { @class = "form-control" })

</div>

</div>

<p>

<div class="form-group">

@Html.Label("Вес, кг", new { @class = "col-md-2 control-label" })

<div class="col-md-10">

@Html.TextBoxFor(m => m.Weight, new { @class = "form-control" })

</div>

</div>

</p>

<p>

<div class="form-group">

@Html.Label("Возраст", new { @class = "col-md-2 control-label" })

<div class="col-md-10">

@Html.TextBoxFor(m => m.Age, new { @class = "form-control" })

</div>

</div>

</p>

<p>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" class="btn btn-default" value="Добавить" />

</div>

</div>

</p>

}

<p><h3>@ViewBag.Message</h3></p>

<div id="results">

@Ajax.ActionLink("Просмотреть список", "Animals", new AjaxOptions { UpdateTargetId = "results", OnSuccess = "OnSuccessAnimals", HttpMethod = "Get", InsertionMode = InsertionMode.Replace })

</div>

</div>

**Arti.cshtml:**

@model IEnumerable<AnimalsEntity.Artiodactyls>

@{

ViewBag.Title = "Arti";

}

<p>

@Ajax.ActionLink("Очистить список", "RemoveArti", new AjaxOptions { OnSuccess = "OnSuccessRemoveArti", HttpMethod = "Delete" })

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.Weight)

</th>

<th>

@Html.DisplayNameFor(model => model.Name)

</th>

<th>

@Html.DisplayNameFor(model => model.Age)

</th>

<th>

@Html.DisplayNameFor(model => model.Cover)

</th>

<th>

@Html.DisplayNameFor(model => model.Temperature)

</th>

<th>

@Html.DisplayNameFor(model => model.IsSwimming)

</th>

<th>

@Html.DisplayNameFor(model => model.IsRum)

</th>

<th>

@Html.DisplayNameFor(model => model.Hoofs)

</th>

<th>

@Html.DisplayNameFor(model => model.HornAvailable)

</th>

<th>

@Html.DisplayNameFor(model => model.HornLength)

</th>

<th></th>

</tr>

@foreach (var item in Model)

{

<tr>

<td>

@Html.DisplayFor(modelItem => item.Weight)

</td>

<td>

@Html.DisplayFor(modelItem => item.Name)

</td>

<td>

@Html.DisplayFor(modelItem => item.Age)

</td>

<td>

@Html.DisplayFor(modelItem => item.Cover)

</td>

<td>

@Html.DisplayFor(modelItem => item.Temperature)

</td>

<td>

@Html.DisplayFor(modelItem => item.IsSwimming)

</td>

<td>

@Html.DisplayFor(modelItem => item.IsRum)

</td>

<td>

@Html.DisplayFor(modelItem => item.Hoofs)

</td>

<td>

@Html.DisplayFor(modelItem => item.HornAvailable)

</td>

<td>

@Html.DisplayFor(modelItem => item.HornLength)

</td>

</tr>

}

</table>

**ArtiView.cshtml:**

@using AnimalsEntity;

@model AnimalsEntity.Artiodactyls

@{

ViewBag.Title = "Добавить парнокопытное";

}

<h2>@ViewBag.Title</h2>

<div>

@using (Ajax.BeginForm("AddArti", new AjaxOptions { OnSuccess = "OnSuccessAddArti", HttpMethod = "Post" }))

{

<hr />

<div class="form-group">

@Html.Label("Наименование", new { @class = "col-md-2 control-label" })

<div class="col-md-10">

@Html.TextBoxFor(m => m.Name, new { @class = "form-control" })

</div>

</div>

<p>

<div class="form-group">

@Html.Label("Количество копыт", new { @class = "col-md-2 control-label" })

<div class="col-md-10">

@Html.TextBoxFor(m => m.Hoofs, new { @class = "form-control" })

</div>

</div>

</p>

<p>

<div class="form-group">

@Html.Label("Животное жвачное? да, нет", new { @class = "col-md-2 control-label" })

<div class="col-md-10">

@Html.CheckBoxFor(m => m.IsRum, new { @class = "form-control" })

</div>

</div>

</p>

<p>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" class="btn btn-default" value="Добавить" />

</div>

</div>

</p>

}

<p><h3>@ViewBag.Message</h3></p>

<div id="arti\_results">

@Ajax.ActionLink("Просмотреть список", "Arti", new AjaxOptions { UpdateTargetId = "arti\_results", OnSuccess = "OnSuccessArti", HttpMethod = "Get", InsertionMode = InsertionMode.Replace })

</div>

</div>

**Birds.cshtml:**

@model IEnumerable<AnimalsEntity.Bird>

@{

ViewBag.Title = "Birds";

}

<p>

@Ajax.ActionLink("Очистить список", "RemoveBirds", new AjaxOptions { OnSuccess = "OnSuccessRemoveBirds", HttpMethod = "Delete" })

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.Weight)

</th>

<th>

@Html.DisplayNameFor(model => model.Name)

</th>

<th>

@Html.DisplayNameFor(model => model.Age)

</th>

<th>

@Html.DisplayNameFor(model => model.IsTalking)

</th>

<th>

@Html.DisplayNameFor(model => model.Wings)

</th>

<th>

@Html.DisplayNameFor(model => model.Eggs)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.Weight)

</td>

<td>

@Html.DisplayFor(modelItem => item.Name)

</td>

<td>

@Html.DisplayFor(modelItem => item.Age)

</td>

<td>

@Html.DisplayFor(modelItem => item.IsTalking)

</td>

<td>

@Html.DisplayFor(modelItem => item.Wings)

</td>

<td>

@Html.DisplayFor(modelItem => item.Eggs)

</td>

</tr>

}

</table>

**BirdsView.cshtml:**

@using AnimalsEntity;

@model AnimalsEntity.Bird

@{

ViewBag.Title = "Добавить птицу";

}

<h2>@ViewBag.Title</h2>

<div>

@using (Ajax.BeginForm("AddBird", new AjaxOptions { OnSuccess = "OnSuccessAddBird", HttpMethod = "Post" }))

{

<hr />

<div class="form-group">

@Html.Label("Наименование", new { @class = "col-md-2 control-label" })

<div class="col-md-10">

@Html.TextBoxFor(m => m.Name, new { @class = "form-control" })

</div>

</div>

<p>

<div class="form-group">

@Html.Label("Окраска крыльев:", new { @class = "col-md-2 control-label" })

<div class="col-md-10">

@Html.TextBoxFor(m => m.Wings, new { @class = "col-md-2 control-label" })

</div>

</div>

</p>

<p>

<div class="form-group">

@Html.Label("Птица говорящая?", new { @class = "col-md-2 control-label" })

<div class="col-md-10">

@Html.CheckBoxFor(m => m.IsTalking, new { @class = "form-control" })

</div>

</div>

</p>

<p>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" class="btn btn-default" value="Добавить" />

</div>

</div>

</p>

}

<p><h3>@ViewBag.Message</h3></p>

<div id="birds\_result">

@Ajax.ActionLink("Просмотреть список", "Birds", new AjaxOptions { UpdateTargetId = "birds\_result", OnSuccess = "OnSuccessBirds", HttpMethod = "Get", InsertionMode = InsertionMode.Replace })

</div>

</div>

**Index.cshtml:**

@{

ViewBag.Title = "Home Page";

}

<div class="jumbotron text-center">

<h1>Мир Животных</h1>

<p class="lead">Веберите ниже нужный пункт</p>

@using (Html.BeginForm("AnimalsView", "Home"))

{

<p><button class="btn btn-primary btn-lg">Животные</button></p>

}

@using (Html.BeginForm("MammalsView", "Home"))

{

<p><button class="btn btn-primary btn-lg">Млекопитающие</button></p>

}

@using (Html.BeginForm("ArtiView", "Home"))

{

<p><button class="btn btn-primary btn-lg">Парнокопытные</button></p>

}

@using (Html.BeginForm("BirdsView", "Home"))

{

<p><button class="btn btn-primary btn-lg">Птицы</button></p>

}

</div>

**Mammals.cshtml:**

@model IEnumerable<AnimalsEntity.Mammal>

<p>

@Ajax.ActionLink("Очистить список", "RemoveMammals", new AjaxOptions { OnSuccess = "OnSuccessRemoveMammals", HttpMethod = "Delete" })

</p>

<div id="mammals">

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.Weight)

</th>

<th>

@Html.DisplayNameFor(model => model.Name)

</th>

<th>

@Html.DisplayNameFor(model => model.Age)

</th>

<th>

@Html.DisplayNameFor(model => model.Cover)

</th>

<th>

@Html.DisplayNameFor(model => model.Temperature)

</th>

<th>

@Html.DisplayNameFor(model => model.IsSwimming)

</th>

<th></th>

</tr>

@foreach (var item in Model)

{

<tr>

<td>

@Html.DisplayFor(modelItem => item.Weight)

</td>

<td>

@Html.DisplayFor(modelItem => item.Name)

</td>

<td>

@Html.DisplayFor(modelItem => item.Age)

</td>

<td>

@Html.DisplayFor(modelItem => item.Cover)

</td>

<td>

@Html.DisplayFor(modelItem => item.Temperature)

</td>

<td>

@Html.DisplayFor(modelItem => item.IsSwimming)

</td>

</tr>

}

</table>

</div>

**MammalsView.cshtml:**

@using AnimalsEntity;

@model AnimalsEntity.Mammal

@{

ViewBag.Title = "Добавить млекопитающее";

}

<h2>@ViewBag.Title</h2>

<div>

@using (Ajax.BeginForm("AddMammal", new AjaxOptions { OnSuccess = "OnSuccessAddMammal", HttpMethod = "Post" }))

{

<hr />

<div class="form-group">

@Html.Label("Наименование", new { @class = "col-md-2 control-label" })

<div class="col-md-10">

@Html.TextBoxFor(m => m.Name, new { @class = "form-control" })

</div>

</div>

<p>

<div class="form-group">

@Html.Label("Температура тела", new { @class = "col-md-2 control-label" })

<div class="col-md-10">

@Html.TextBoxFor(m => m.Temperature, new { @class = "col-md-2 control-label" })

</div>

</div>

</p>

<p>

<div class="form-group">

@Html.Label("Животное умеет плавать?", new { @class = "col-md-2 control-label" })

<div class="col-md-10">

@Html.CheckBoxFor(m => m.IsSwimming, new { @class = "form-control" })

</div>

</div>

</p>

<p>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" class="btn btn-default" value="Добавить" />

</div>

</div>

</p>

}

<p><h3>@ViewBag.Message</h3></p>

<div id="mammal\_result">

@Ajax.ActionLink("Просмотреть список", "Mammals", new AjaxOptions { UpdateTargetId = "mammal\_result", OnSuccess = "OnSuccessMammals", HttpMethod = "Get", InsertionMode = InsertionMode.Replace })

</div>

</div>

# **Лабораторная работа №5**

Вариант 13

*Задание*: Реализовать веб-сервис и клиентское приложение (по аналогии с лабораторной работой № 3). Веб-сервис должен быть реализован с использованием технологии Windows communication Foundation (WCF). Обращение должно производиться к веб-сервису, разработанному студентом с любым другим вариантом задания.

*Результат выполнения программы:*

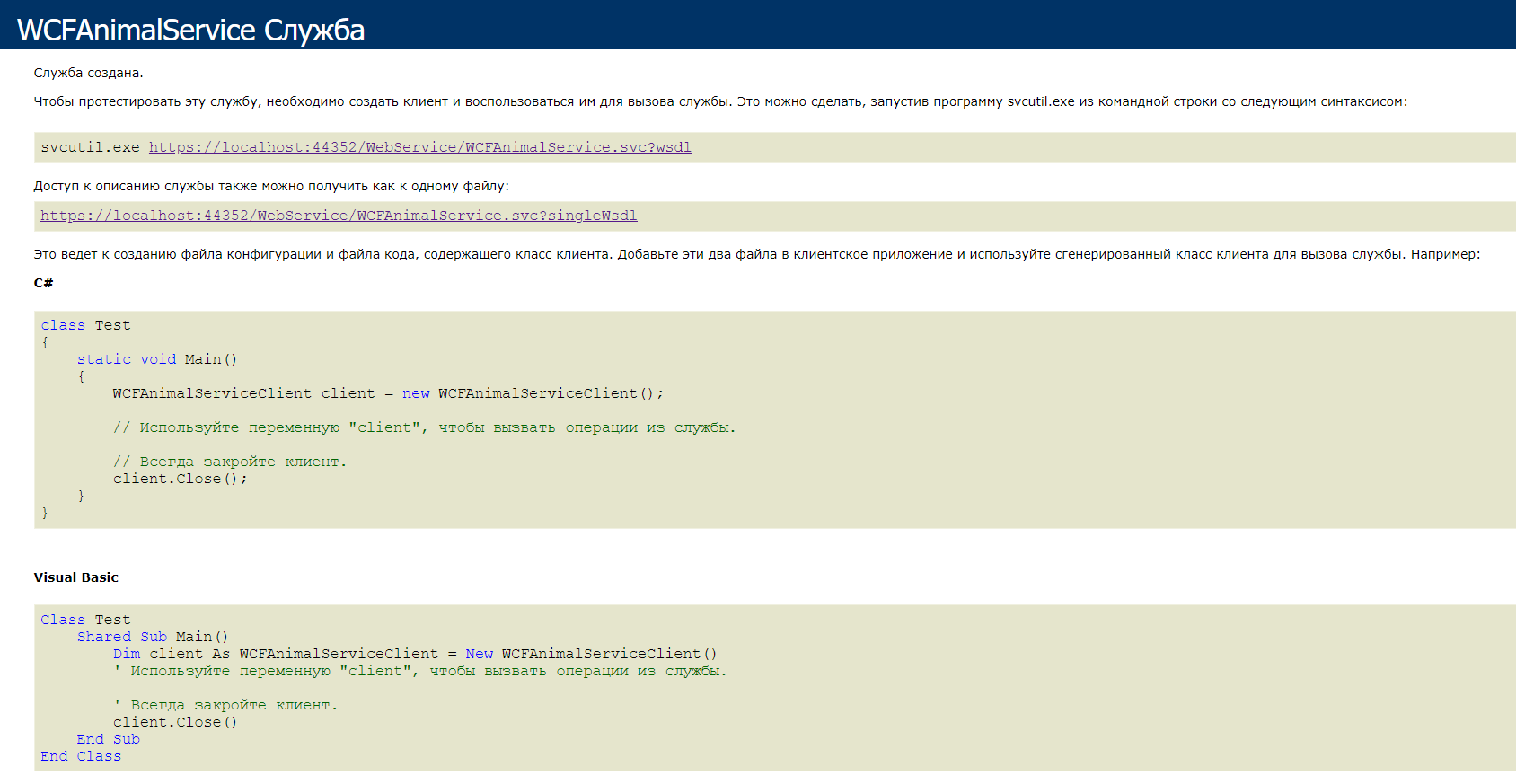


Рис.28 – Вид созданной WCF службы

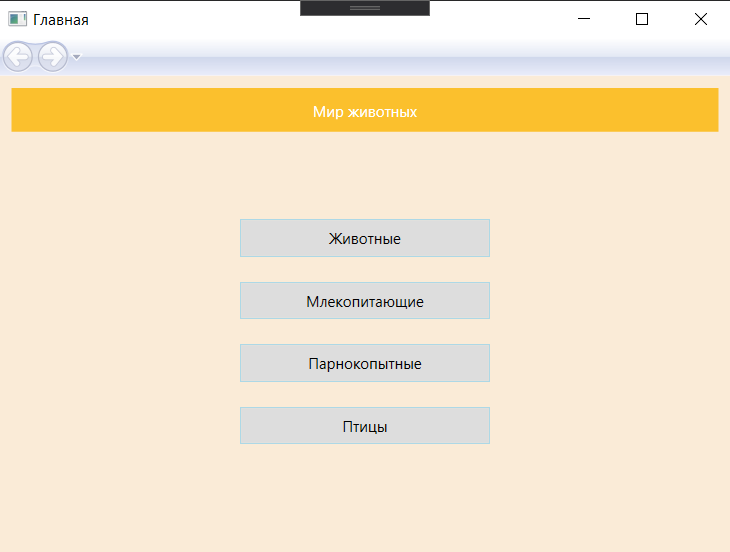


Рис. 29 – Главное меню

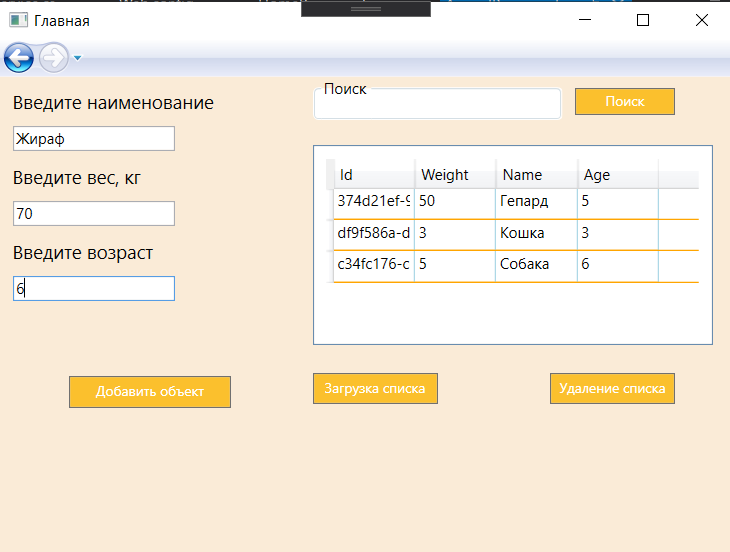


Рис. 30 – Оконное приложение, работающее с веб-службой

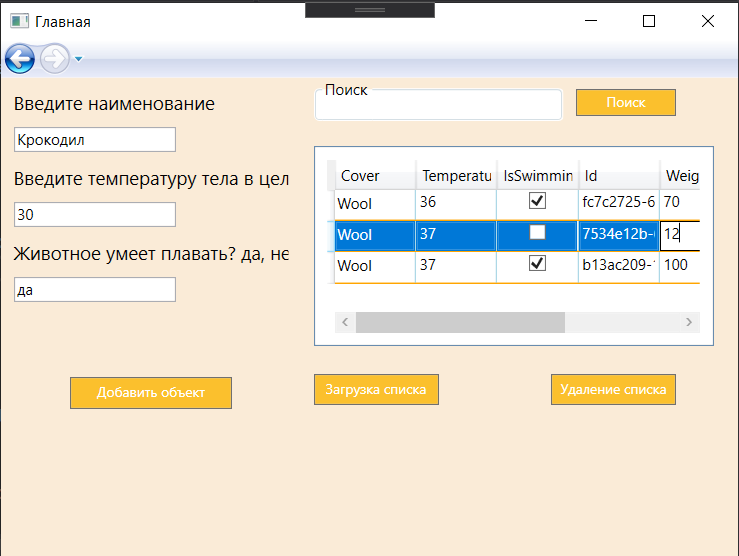


Рис. 31 – Оконное приложение, работающее с веб-службой

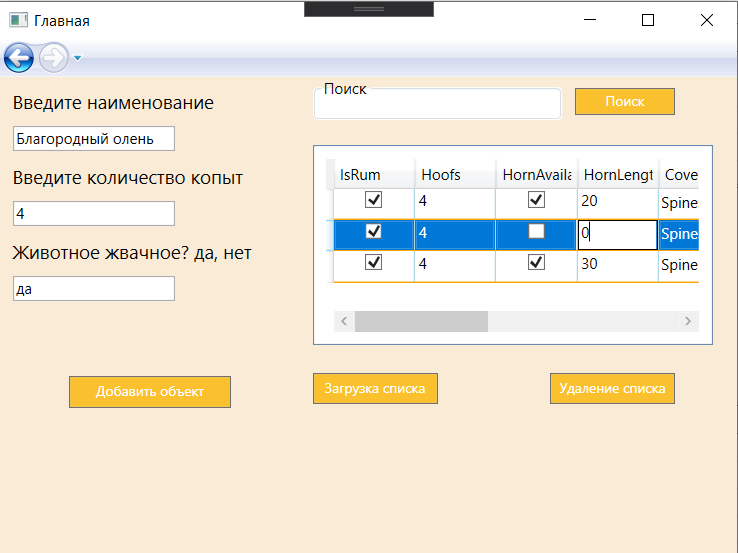


Рис. 32 – Оконное приложение, работающее с веб-службой

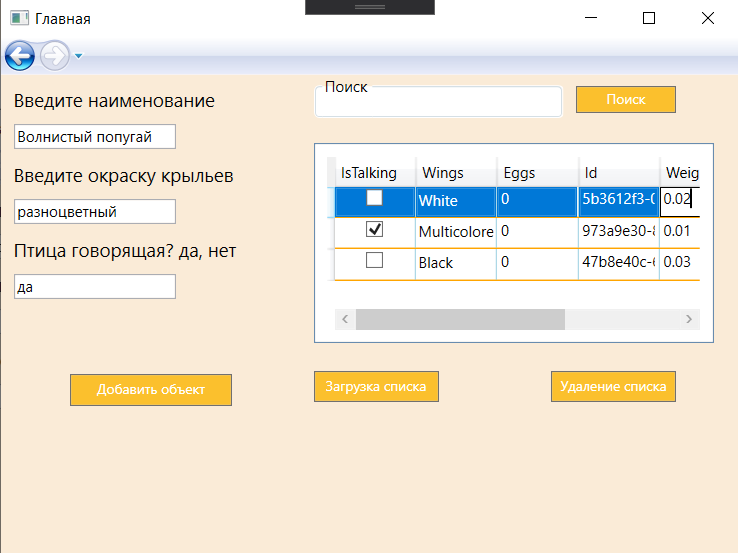


Рис. 33 – Оконное приложение, работающее с веб-службой

*Листинг программных модулей:*

**IWCFAnimalService.cs:**

using AnimalsEntity;

using System.Collections.Generic;

using System.ServiceModel;

namespace WebAnimalProject.WebService

{

// ПРИМЕЧАНИЕ. Команду "Переименовать" в меню "Рефакторинг" можно использовать для одновременного изменения имени интерфейса "IWCFAnimalService" в коде и файле конфигурации.

[ServiceContract]

public interface IWCFAnimalService

{

[OperationContract]

bool AddAnimal(string name, int age, double weight);

[OperationContract]

bool AddMammal(string name, double temp, bool isIswimming);

[OperationContract]

bool AddAtri(string name, int h\_count, bool isRum);

[OperationContract]

bool AddBird(string name, Color wings, bool isTalking);

[OperationContract]

List<Animal> GetAnimals();

[OperationContract]

List<Mammal> GetMammals();

[OperationContract]

List<Artiodactyls> GetArti();

[OperationContract]

List<Bird> GetBirds();

[OperationContract]

bool RemoveAnimals();

[OperationContract]

bool RemoveMammals();

[OperationContract]

bool RemoveArti();

[OperationContract]

bool RemoveBirds();

}

}

**WCFAnimalService.svc.cs:**

using AnimalsEntity;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.Text;

using WebAnimalProject.Context;

namespace WebAnimalProject.WebService

{

// ПРИМЕЧАНИЕ. Команду "Переименовать" в меню "Рефакторинг" можно использовать для одновременного изменения имени класса "WCFAnimalService" в коде, SVC-файле и файле конфигурации.

// ПРИМЕЧАНИЕ. Чтобы запустить клиент проверки WCF для тестирования службы, выберите элементы WCFAnimalService.svc или WCFAnimalService.svc.cs в обозревателе решений и начните отладку.

public class WCFAnimalService : IWCFAnimalService

{

public bool AddAnimal(string name, int age, double weight)

{

using (AnimalContext bd = new AnimalContext())

{

Animal animal = new Animal

{

Age = age,

Name = name,

Weight = weight

};

bd.Animals.Add(animal);

bd.SaveChanges();

return true;

}

}

public bool AddMammal(string name, double temp, bool isIswimming)

{

using (AnimalContext bd = new AnimalContext())

{

Mammal animal = new Mammal

{

IsSwimming = isIswimming,

Name = name,

Temperature = temp

};

bd.Mammals.Add(animal);

bd.SaveChanges();

return true;

}

}

public bool AddAtri(string name, int h\_count, bool isRum)

{

using (AnimalContext bd = new AnimalContext())

{

Artiodactyls animal = new Artiodactyls

{

IsRum = isRum,

Name = name,

Hoofs = h\_count

};

bd.Artiodactylses.Add(animal);

bd.SaveChanges();

return true;

}

}

public bool AddBird(string name, Color wings, bool isTalking)

{

using (AnimalContext bd = new AnimalContext())

{

Bird animal = new Bird

{

Wings = wings,

Name = name,

IsTalking = isTalking

};

bd.Birds.Add(animal);

bd.SaveChanges();

return true;

}

}

public List<Animal> GetAnimals()

{

List<Animal> animals = new List<Animal>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Animals.ToList();

animals.AddRange(founded\_animal);

}

return animals;

}

public List<Mammal> GetMammals()

{

List<Mammal> animals = new List<Mammal>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Mammals.ToList();

animals.AddRange(founded\_animal);

}

return animals;

}

public List<Artiodactyls> GetArti()

{

List<Artiodactyls> animals = new List<Artiodactyls>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Artiodactylses.ToList();

animals.AddRange(founded\_animal);

}

return animals;

}

public List<Bird> GetBirds()

{

List<Bird> animals = new List<Bird>();

using (AnimalContext bd = new AnimalContext())

{

var founded\_animal = bd.Birds.ToList();

animals.AddRange(founded\_animal);

}

return animals;

}

public bool RemoveAnimals()

{

using (AnimalContext bd = new AnimalContext())

{

foreach (var item in bd.Animals)

{

bd.Animals.Remove(item);

}

bd.SaveChanges();

}

return true;

}

public bool RemoveMammals()

{

using (AnimalContext bd = new AnimalContext())

{

foreach (var item in bd.Mammals)

{

bd.Mammals.Remove(item);

}

bd.SaveChanges();

}

return true;

}

public bool RemoveArti()

{

using (AnimalContext bd = new AnimalContext())

{

foreach (var item in bd.Artiodactylses)

{

bd.Artiodactylses.Remove(item);

}

bd.SaveChanges();

}

return true;

}

public bool RemoveBirds()

{

using (AnimalContext bd = new AnimalContext())

{

foreach (var item in bd.Birds)

{

bd.Birds.Remove(item);

}

bd.SaveChanges();

}

return true;

}

}

}

**WCFTestPage.xaml:**

<Page x:Class="WpfClientApp.WCFTestPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:WpfClientApp"

mc:Ignorable="d"

Background="AntiqueWhite"

d:DesignHeight="300" d:DesignWidth="580"

Title="AnimalPage">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="240"/>

<ColumnDefinition Width="auto"/>

</Grid.ColumnDefinitions>

<StackPanel Orientation="Vertical" Margin="10" Grid.Column="0">

<TextBlock x:Name="txt\_first" FontSize="15" Text=""></TextBlock>

<TextBox Name="edit\_first" Width="130" HorizontalAlignment="Left" Height="20" Margin="0,10,0,0"></TextBox>

<TextBlock x:Name="txt\_second" Margin="0,10,0,0" FontSize="15" Text=""></TextBlock>

<TextBox Name="edit\_second" Width="130" HorizontalAlignment="Left" Height="20" Margin="0,10,0,0"></TextBox>

<TextBlock x:Name="txt\_third" Margin="0,10,0,0" FontSize="15" Text=""></TextBlock>

<TextBox Name="edit\_third" Width="130" HorizontalAlignment="Left" Height="20" Margin="0,10,0,0"></TextBox>

<Button Width="130" FontFamily="bold" FontWeight="Regular"

Padding="3" Height="25" Background="#FBC02D" Click="Button\_Add"

Foreground="White" Margin="0,60,0,0" FontSize="11">

Добавить объект

</Button>

</StackPanel>

<StackPanel Orientation="Vertical" Grid.Column="1">

<StackPanel Orientation="Horizontal">

<GroupBox Background="White" HorizontalAlignment="Left" Width="200" Header="Поиск" Margin="10,0,0,0" Height="35">

<TextBox Name="edit\_search" Foreground="Green" BorderBrush="White" Height="35" FontSize="10">

</TextBox>

</GroupBox>

<Button FontFamily="bold" FontWeight="Regular"

Margin="10,5,0,0" Width="80" Click="Button\_Search"

Padding="3" Height="22" Background="#FBC02D"

Foreground="White" FontSize="11">

Поиск

</Button>

</StackPanel>

<DataGrid BorderThickness="1"

Padding="10" Margin="10,20,0,0"

ColumnWidth="65" HeadersVisibility="All"

GridLinesVisibility="All" HorizontalGridLinesBrush="Orange"

VerticalGridLinesBrush="LightBlue" AutoGenerateColumns="true"

Height="160" VerticalAlignment="Top" Background="White"

AreRowDetailsFrozen="True" Width="320"

SelectionMode="Single" RowHeight="25" CanUserAddRows="False"

CanUserDeleteRows="False" ItemsSource="{Binding}"

Name="tb\_list" CanUserSortColumns = "False">

</DataGrid>

<StackPanel Orientation="Horizontal" Margin="0,17,0,0" HorizontalAlignment="Left">

<Button FontFamily="bold" FontWeight="Regular"

Margin="10,5,0,0" Width="100" Click="Button\_Load"

Padding="3" Height="25" Background="#FBC02D"

Foreground="White" FontSize="11">

Загрузка списка

</Button>

<Button FontFamily="bold" FontWeight="Regular"

Margin="90,5,0,0" Width="100" Click="Button\_Delete"

Padding="3" Height="25" Background="#FBC02D"

Foreground="White" FontSize="11">

Удаление списка

</Button>

</StackPanel>

</StackPanel>

</Grid>

</Page>

**WCFTestPage.xaml.cs:**

using AnimalsEntity;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Linq;

using System.ServiceModel.Channels;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using static WpfClientApp.HomePage;

namespace WpfClientApp

{

/// <summary>

/// Логика взаимодействия для WCFTestPage.xaml

/// </summary>

public partial class WCFTestPage : Page

{

AnimalType type;

AnimalServiceReference.WCFAnimalServiceClient client = new AnimalServiceReference.WCFAnimalServiceClient();

public List<Animal> Animals = new List<Animal>();

public List<Artiodactyls> Artiodactyls = new List<Artiodactyls>();

public List<Mammal> Mammals = new List<Mammal>();

public List<Bird> Birds = new List<Bird>();

public WCFTestPage()

{

InitializeComponent();

var parentWindow = this.Parent as Window;

NavigationService nav = NavigationService.GetNavigationService(this);

//if (parentWindow != null)

// parentWindow.Loaded += ParentWindow\_Loaded;

//this.NavigationService.LoadCompleted += NavigationService\_LoadCompleted;

type = StaticAnimal.TypeOfAnimal;

switch (type)

{

case AnimalType.Animal:

{

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите вес, кг";

txt\_third.Text = "Введите возраст";

}

break;

case AnimalType.Mammal:

{

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите температуру тела в цельсиях";

txt\_third.Text = "Животное умеет плавать? да, нет";

}

break;

case AnimalType.Anti:

{

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите количество копыт";

txt\_third.Text = "Животное жвачное? да, нет";

}

break;

case AnimalType.Bird:

{

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите окраску крыльев";

txt\_third.Text = "Птица говорящая? да, нет";

}

break;

default:

break;

}

}

private void Button\_Add(object sender, RoutedEventArgs e)

{

AddInfo();

}

private void AddInfo()

{

if (string.IsNullOrWhiteSpace(txt\_first.Text) || string.IsNullOrWhiteSpace(txt\_second.Text)

|| string.IsNullOrWhiteSpace(txt\_third.Text))

{

MessageBox.Show("Введите все данные");

return;

}

switch (type)

{

case AnimalType.Animal:

{

Animal animal = new Animal();

animal.Age = int.Parse(edit\_third.Text);

animal.Name = edit\_first.Text;

animal.Weight = int.Parse(edit\_second.Text);

Animals.Add(animal);

client.AddAnimal(animal.Name, animal.Age, animal.Weight);

Animals = client.GetAnimals().ToList();

tb\_list.DataContext = Animals.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

case AnimalType.Mammal:

{

Mammal animal = new Mammal();

animal.Temperature = int.Parse(edit\_second.Text);

animal.Name = edit\_first.Text;

animal.IsSwimming = (edit\_third.Text == "да") ? true : false;

Mammals.Add(animal);

client.AddMammal(animal.Name, animal.Temperature, animal.IsSwimming);

Mammals = client.GetMammals().ToList();

tb\_list.DataContext = Mammals.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

case AnimalType.Anti:

{

Artiodactyls animal = new Artiodactyls();

animal.IsRum = (edit\_third.Text == "да") ? true : false;

animal.Name = edit\_first.Text;

animal.Hoofs = int.Parse(edit\_second.Text);

Artiodactyls.Add(animal);

client.AddAtri(animal.Name, animal.Hoofs, animal.IsRum);

Artiodactyls = client.GetArti().ToList();

tb\_list.DataContext = Artiodactyls.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

case AnimalType.Bird:

{

Bird animal = new Bird();

animal.Wings = (edit\_second.Text == "черный") ? AnimalsEntity.Color.Black : (edit\_second.Text == "белый") ? AnimalsEntity.Color.White :

(edit\_second.Text == "разноцветный") ? AnimalsEntity.Color.Multicolored : AnimalsEntity.Color.Black;

animal.Name = edit\_first.Text;

animal.IsTalking = (edit\_third.Text == "да") ? true : false;

Birds.Add(animal);

client.AddBird(animal.Name, animal.Wings, animal.IsTalking);

Birds = client.GetBirds().ToList();

tb\_list.DataContext = Birds.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

default:

break;

}

}

private void Button\_Load(object sender, RoutedEventArgs e)

{

LoadInfo();

}

private void LoadInfo()

{

switch (type)

{

case AnimalType.Animal:

{

Animals = client.GetAnimals().ToList();

tb\_list.DataContext = Animals.ToList();

}

break;

case AnimalType.Mammal:

{

Mammals = client.GetMammals().ToList();

tb\_list.DataContext = Mammals.ToList();

}

break;

case AnimalType.Anti:

{

Artiodactyls = client.GetArti().ToList();

tb\_list.DataContext = Artiodactyls.ToList();

}

break;

case AnimalType.Bird:

{

Birds = client.GetBirds().ToList();

tb\_list.DataContext = Birds.ToList();

}

break;

default:

break;

}

}

private void Button\_Search(object sender, RoutedEventArgs e)

{

switch (type)

{

case AnimalType.Animal:

{

BindingList<Animal> BS = new BindingList<Animal>(Animals.Where(m => m.Age.ToString().Contains(edit\_search.Text) || m.Name.ToLower().Contains(edit\_search.Text.ToLower()) || m.Weight.ToString().Contains(edit\_search.Text)).ToList());

tb\_list.DataContext = BS;

}

break;

case AnimalType.Mammal:

{

BindingList<Mammal> BS = new BindingList<Mammal>(Mammals.Where(m => m.Name.ToLower().Contains(edit\_search.Text.ToLower())).ToList());

tb\_list.DataContext = BS;

}

break;

case AnimalType.Anti:

{

BindingList<Artiodactyls> BS = new BindingList<Artiodactyls>(Artiodactyls.Where(m => m.Name.ToLower().Contains(edit\_search.Text.ToLower())).ToList());

tb\_list.DataContext = BS;

}

break;

case AnimalType.Bird:

{

BindingList<Bird> BS = new BindingList<Bird>(Birds.Where(m => m.Name.ToLower().Contains(edit\_search.Text.ToLower())).ToList());

tb\_list.DataContext = BS;

}

break;

default:

break;

}

}

private void Button\_Delete(object sender, RoutedEventArgs e)

{

switch (type)

{

case AnimalType.Animal:

{

Animals = new List<Animal>();

bool result = client.RemoveAnimals();

if (result)

MessageBox.Show("Успешно!");

else

MessageBox.Show("Что-то пошло не так..");

tb\_list.DataContext = Animals.ToList();

}

break;

case AnimalType.Mammal:

{

Mammals = new List<Mammal>();

bool result = client.RemoveMammals();

if (result)

MessageBox.Show("Успешно!");

else

MessageBox.Show("Что-то пошло не так..");

tb\_list.DataContext = Mammals.ToList();

}

break;

case AnimalType.Anti:

{

Artiodactyls = new List<Artiodactyls>();

bool result = client.RemoveArti();

if (result)

MessageBox.Show("Успешно!");

else

MessageBox.Show("Что-то пошло не так..");

tb\_list.DataContext = Artiodactyls.ToList();

}

break;

case AnimalType.Bird:

{

Birds = new List<Bird>();

bool result = client.RemoveBirds();

if (result)

MessageBox.Show("Успешно!");

else

MessageBox.Show("Что-то пошло не так..");

tb\_list.DataContext = Birds.ToList();

}

break;

default:

break;

}

}

}

}

# **Лабораторная работа №6**

Вариант 13

*Задание*: Реализовать создание объектов классов, реализованных в лабораторной работе 1 с помощью механизмов Windows Workflow Foundation (WWF). Так же в завершении процессов объекты должны сохраняться в файл (сериализация).

*Результат выполнения программы:*

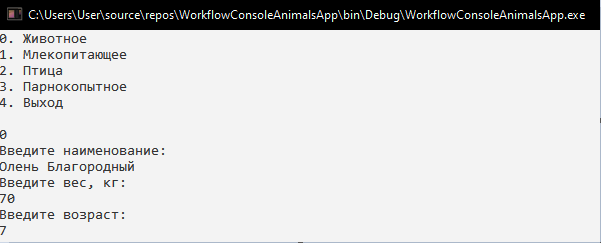
**

Рис. 34 – Добавление животного



Рис. 35 – Добавление млекопитающего

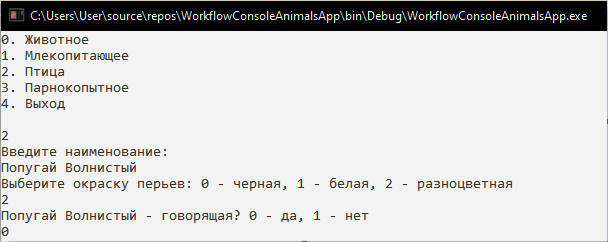


Рис. 36 – Добавление птицы

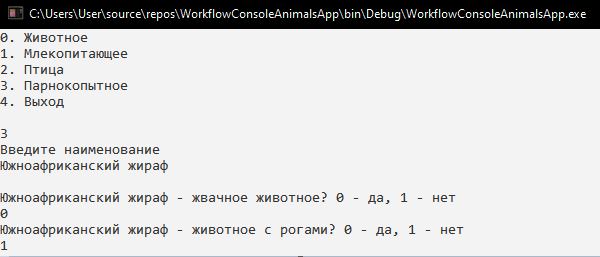


Рис. 37 – Добавление парнокопытного

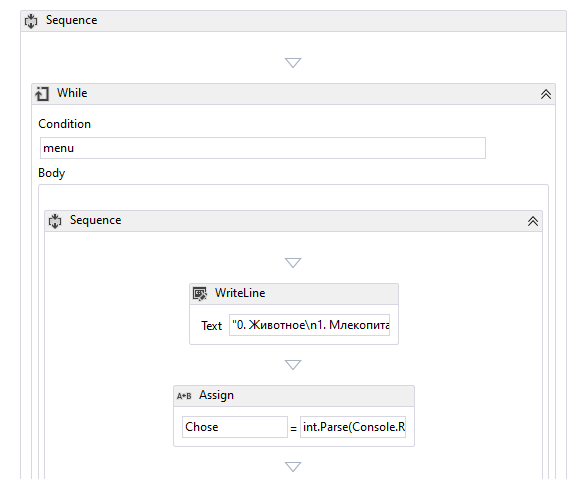
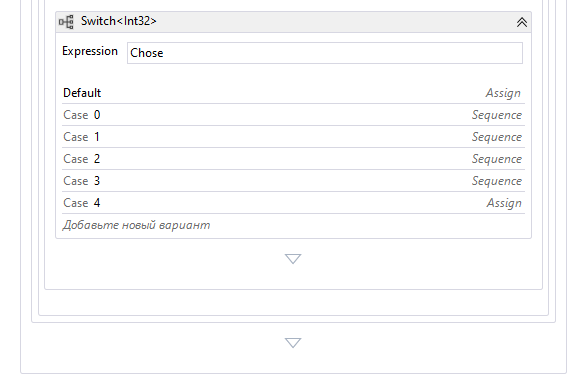
 

Рис. 38 – Рабочий процесс

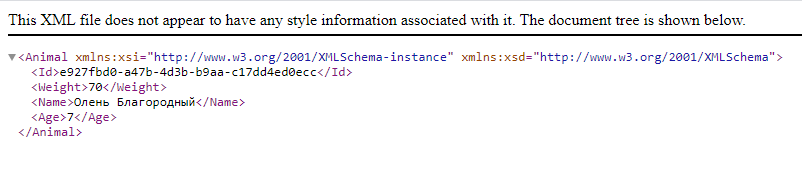


Рис. 39 – Обзор сериализованного объекта (животное)

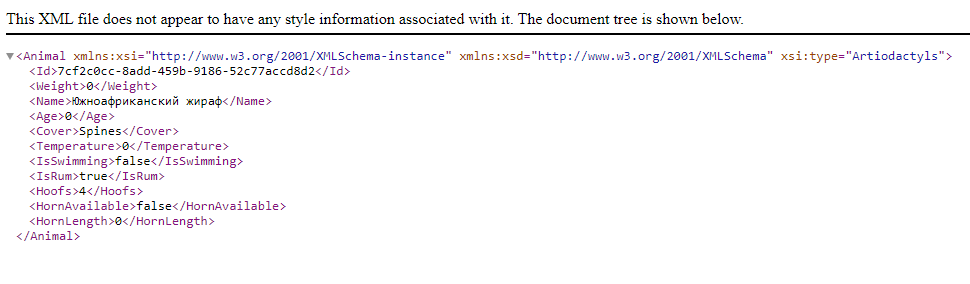


Рис. 40 – Обзор сериализованного объекта (парнокопытное)

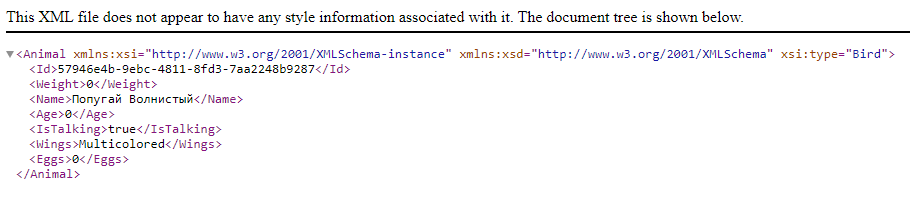


Рис. 41 – Обзор сериализованного объекта (птица)

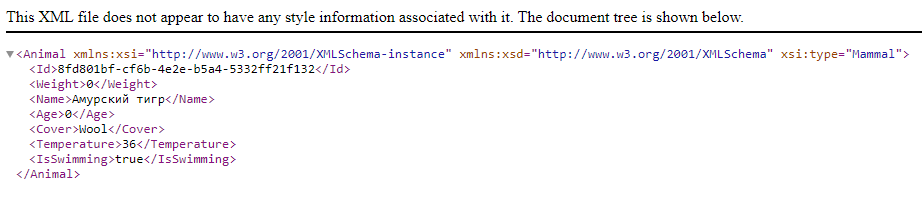


Рис. 42 – Обзор сериализованного объекта (млекопитающее)

*Листинг программных модулей:*

**Workflow1.xaml:**

<Activity mc:Ignorable="sap sap2010 sads" x:Class="WorkflowConsoleAnimalsApp.Workflow1" sap2010:ExpressionActivityEditor.ExpressionActivityEditor="C#" sap2010:WorkflowViewState.IdRef="WorkflowConsoleAnimalsApp.Workflow1\_1"

xmlns="http://schemas.microsoft.com/netfx/2009/xaml/activities"

xmlns:a="clr-namespace:AnimalsEntity;assembly=AnimalsEntity"

xmlns:local="clr-namespace:WorkflowConsoleAnimalsApp"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:mca="clr-namespace:Microsoft.CSharp.Activities;assembly=System.Activities"

xmlns:sads="http://schemas.microsoft.com/netfx/2010/xaml/activities/debugger"

xmlns:sap="http://schemas.microsoft.com/netfx/2009/xaml/activities/presentation"

xmlns:sap2010="http://schemas.microsoft.com/netfx/2010/xaml/activities/presentation"

xmlns:scg="clr-namespace:System.Collections.Generic;assembly=mscorlib"

xmlns:sco="clr-namespace:System.Collections.ObjectModel;assembly=mscorlib"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">

<TextExpression.NamespacesForImplementation>

<sco:Collection x:TypeArguments="x:String">

<x:String>System</x:String>

<x:String>System.Collections.Generic</x:String>

<x:String>System.Data</x:String>

<x:String>System.Linq</x:String>

<x:String>System.Text</x:String>

<x:String>AnimalsEntity</x:String>

</sco:Collection>

</TextExpression.NamespacesForImplementation>

<TextExpression.ReferencesForImplementation>

<sco:Collection x:TypeArguments="AssemblyReference">

<AssemblyReference>Microsoft.CSharp</AssemblyReference>

<AssemblyReference>PresentationCore</AssemblyReference>

<AssemblyReference>PresentationFramework</AssemblyReference>

<AssemblyReference>System</AssemblyReference>

<AssemblyReference>System.Activities</AssemblyReference>

<AssemblyReference>System.Activities.Presentation</AssemblyReference>

<AssemblyReference>System.Core</AssemblyReference>

<AssemblyReference>System.Data</AssemblyReference>

<AssemblyReference>System.Runtime.Serialization</AssemblyReference>

<AssemblyReference>System.ServiceModel</AssemblyReference>

<AssemblyReference>System.ServiceModel.Activities</AssemblyReference>

<AssemblyReference>System.Xaml</AssemblyReference>

<AssemblyReference>System.Xml</AssemblyReference>

<AssemblyReference>System.Xml.Linq</AssemblyReference>

<AssemblyReference>WindowsBase</AssemblyReference>

<AssemblyReference>AnimalsEntity</AssemblyReference>

<AssemblyReference>mscorlib</AssemblyReference>

<AssemblyReference>WorkflowConsoleAnimalsApp</AssemblyReference>

</sco:Collection>

</TextExpression.ReferencesForImplementation>

<Sequence sap2010:WorkflowViewState.IdRef="Sequence\_2">

<Sequence.Variables>

<Variable x:TypeArguments="x:Int32" Name="Chose" />

<Variable x:TypeArguments="x:String" Name="value" />

<Variable x:TypeArguments="x:Boolean" Name="res" />

<Variable x:TypeArguments="x:Int32" Name="choice" />

<Variable x:TypeArguments="x:String" Name="Name" />

<Variable x:TypeArguments="x:Boolean" Name="IsRum" />

<Variable x:TypeArguments="x:String" Name="variable1" />

<Variable x:TypeArguments="x:Boolean" Name="IsHorn" />

<Variable x:TypeArguments="x:String" Name="variable2" />

<Variable x:TypeArguments="x:Boolean" Default="True" Name="menu" />

<Variable x:TypeArguments="x:String" Name="variable3" />

</Sequence.Variables>

<While sap2010:WorkflowViewState.IdRef="While\_1">

<While.Condition>

<mca:CSharpValue x:TypeArguments="x:Boolean">menu</mca:CSharpValue>

</While.Condition>

<Sequence sap2010:WorkflowViewState.IdRef="Sequence\_6">

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_11">

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">"0. Животное\n1. Млекопитающее\n2. Птица\n3. Парнокопытное\n4. Выход\n"</mca:CSharpValue>

</InArgument>

</WriteLine>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_24">

<Assign.To>

<OutArgument x:TypeArguments="x:Int32">

<mca:CSharpReference x:TypeArguments="x:Int32">Chose</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:Int32">

<mca:CSharpValue x:TypeArguments="x:Int32">int.Parse(Console.ReadLine())</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Switch x:TypeArguments="x:Int32" sap2010:WorkflowViewState.IdRef="Switch`1\_1">

<Switch.Default>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_32">

<Assign.To>

<OutArgument x:TypeArguments="x:Boolean">

<mca:CSharpReference x:TypeArguments="x:Boolean">menu</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:Boolean">False</InArgument>

</Assign.Value>

</Assign>

</Switch.Default>

<Switch.Expression>

<InArgument x:TypeArguments="x:Int32">

<mca:CSharpValue x:TypeArguments="x:Int32">Chose</mca:CSharpValue>

</InArgument>

</Switch.Expression>

<Sequence x:Key="0" sap2010:WorkflowViewState.IdRef="Sequence\_1">

<Sequence.Variables>

<Variable x:TypeArguments="x:Double" Name="Weight" />

<Variable x:TypeArguments="x:Int32" Name="age" />

<Variable x:TypeArguments="x:String" Name="Name" />

<Variable x:TypeArguments="a:Animal" Name="animal" />

</Sequence.Variables>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_1" Text="Введите наименование:" />

<Assign sap2010:WorkflowViewState.IdRef="Assign\_1">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">Name</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_2" Text="Введите вес, кг:" />

<Assign sap2010:WorkflowViewState.IdRef="Assign\_2">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">value</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_3">

<Assign.To>

<OutArgument x:TypeArguments="x:Double">

<mca:CSharpReference x:TypeArguments="x:Double">Weight</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:Double">

<mca:CSharpValue x:TypeArguments="x:Double">double.Parse(value)</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_3" Text="Введите возраст: " />

<Assign sap2010:WorkflowViewState.IdRef="Assign\_4">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">value</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_5">

<Assign.To>

<OutArgument x:TypeArguments="x:Int32">

<mca:CSharpReference x:TypeArguments="x:Int32">age</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:Int32">

<mca:CSharpValue x:TypeArguments="x:Int32">int.Parse(value)</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<local:AnimalActivity sap2010:WorkflowViewState.IdRef="AnimalActivity\_1">

<local:AnimalActivity.Age>

<InArgument x:TypeArguments="x:Int32">

<mca:CSharpValue x:TypeArguments="x:Int32">age</mca:CSharpValue>

</InArgument>

</local:AnimalActivity.Age>

<local:AnimalActivity.Name>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Name</mca:CSharpValue>

</InArgument>

</local:AnimalActivity.Name>

<local:AnimalActivity.OutAnimal>

<OutArgument x:TypeArguments="a:Animal">

<mca:CSharpReference x:TypeArguments="a:Animal">animal</mca:CSharpReference>

</OutArgument>

</local:AnimalActivity.OutAnimal>

<local:AnimalActivity.Weight>

<InArgument x:TypeArguments="x:Double">

<mca:CSharpValue x:TypeArguments="x:Double">Weight</mca:CSharpValue>

</InArgument>

</local:AnimalActivity.Weight>

</local:AnimalActivity>

<local:XMLDataSeedActivity x:TypeArguments="a:Animal" FileData="animal.xml" sap2010:WorkflowViewState.IdRef="XMLDataSeedActivity`1\_1">

<local:XMLDataSeedActivity.Object>

<InArgument x:TypeArguments="a:Animal">

<mca:CSharpValue x:TypeArguments="a:Animal">animal</mca:CSharpValue>

</InArgument>

</local:XMLDataSeedActivity.Object>

</local:XMLDataSeedActivity>

</Sequence>

<Sequence x:Key="1" sap2010:WorkflowViewState.IdRef="Sequence\_3">

<Sequence.Variables>

<Variable x:TypeArguments="a:TypeCover" Name="Cover" />

<Variable x:TypeArguments="x:Double" Name="Temperature" />

<Variable x:TypeArguments="x:Boolean" Name="IsSwimming" />

<Variable x:TypeArguments="x:String" Name="Name" />

<Variable x:TypeArguments="x:Int32" Name="choice" />

<Variable x:TypeArguments="a:Mammal" Name="mammal" />

</Sequence.Variables>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_4" Text="Введите наименование:" />

<Assign sap2010:WorkflowViewState.IdRef="Assign\_6">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">Name</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_5" Text="Выберите тип покрова: 0 - иглы, 1 - шерсть, 2 - панцирь" />

<Assign sap2010:WorkflowViewState.IdRef="Assign\_7">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">value</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_8">

<Assign.To>

<OutArgument x:TypeArguments="x:Int32">

<mca:CSharpReference x:TypeArguments="x:Int32">choice</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:Int32">

<mca:CSharpValue x:TypeArguments="x:Int32">int.Parse(value)</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Switch x:TypeArguments="x:Int32" sap2010:WorkflowViewState.IdRef="Switch`1\_2">

<Switch.Expression>

<InArgument x:TypeArguments="x:Int32">

<mca:CSharpValue x:TypeArguments="x:Int32">choice</mca:CSharpValue>

</InArgument>

</Switch.Expression>

<Assign x:Key="0" sap2010:WorkflowViewState.IdRef="Assign\_9">

<Assign.To>

<OutArgument x:TypeArguments="a:TypeCover">

<mca:CSharpReference x:TypeArguments="a:TypeCover">Cover</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="a:TypeCover">

<mca:CSharpValue x:TypeArguments="a:TypeCover">TypeCover.Spines</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign x:Key="1" sap2010:WorkflowViewState.IdRef="Assign\_10">

<Assign.To>

<OutArgument x:TypeArguments="a:TypeCover">

<mca:CSharpReference x:TypeArguments="a:TypeCover">Cover</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="a:TypeCover">

<mca:CSharpValue x:TypeArguments="a:TypeCover">TypeCover.Wool</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign x:Key="2" sap2010:WorkflowViewState.IdRef="Assign\_11">

<Assign.To>

<OutArgument x:TypeArguments="a:TypeCover">

<mca:CSharpReference x:TypeArguments="a:TypeCover">Cover</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="a:TypeCover">

<mca:CSharpValue x:TypeArguments="a:TypeCover">TypeCover.Shell</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

</Switch>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_6" Text="Введите температуру тела:" />

<Assign sap2010:WorkflowViewState.IdRef="Assign\_12">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">value</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_13">

<Assign.To>

<OutArgument x:TypeArguments="x:Double">

<mca:CSharpReference x:TypeArguments="x:Double">Temperature</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:Double">

<mca:CSharpValue x:TypeArguments="x:Double">double.Parse(value)</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_7">

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">$"{Name} плавает? 0 - да, 1 - нет"</mca:CSharpValue>

</InArgument>

</WriteLine>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_14">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">value</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_15">

<Assign.To>

<OutArgument x:TypeArguments="x:Boolean">

<mca:CSharpReference x:TypeArguments="x:Boolean">IsSwimming</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:Boolean">

<mca:CSharpValue x:TypeArguments="x:Boolean">(value == "0") ? true : false</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<local:MammalActivity sap2010:WorkflowViewState.IdRef="MammalActivity\_1">

<local:MammalActivity.Cover>

<InArgument x:TypeArguments="a:TypeCover">

<mca:CSharpValue x:TypeArguments="a:TypeCover">Cover</mca:CSharpValue>

</InArgument>

</local:MammalActivity.Cover>

<local:MammalActivity.IsSwimming>

<InArgument x:TypeArguments="x:Boolean">

<mca:CSharpValue x:TypeArguments="x:Boolean">IsSwimming</mca:CSharpValue>

</InArgument>

</local:MammalActivity.IsSwimming>

<local:MammalActivity.Name>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Name</mca:CSharpValue>

</InArgument>

</local:MammalActivity.Name>

<local:MammalActivity.OutAnimal>

<OutArgument x:TypeArguments="a:Mammal">

<mca:CSharpReference x:TypeArguments="a:Mammal">mammal</mca:CSharpReference>

</OutArgument>

</local:MammalActivity.OutAnimal>

<local:MammalActivity.Temperature>

<InArgument x:TypeArguments="x:Double">

<mca:CSharpValue x:TypeArguments="x:Double">Temperature</mca:CSharpValue>

</InArgument>

</local:MammalActivity.Temperature>

</local:MammalActivity>

<local:XMLDataSeedActivity x:TypeArguments="a:Mammal" FileData="mammal.xml" sap2010:WorkflowViewState.IdRef="XMLDataSeedActivity`1\_2">

<local:XMLDataSeedActivity.Object>

<InArgument x:TypeArguments="a:Mammal">

<mca:CSharpValue x:TypeArguments="a:Mammal">mammal</mca:CSharpValue>

</InArgument>

</local:XMLDataSeedActivity.Object>

</local:XMLDataSeedActivity>

</Sequence>

<Sequence x:Key="2" sap2010:WorkflowViewState.IdRef="Sequence\_4">

<Sequence.Variables>

<Variable x:TypeArguments="x:String" Name="Name" />

<Variable x:TypeArguments="a:Color" Name="Wings" />

<Variable x:TypeArguments="x:Boolean" Name="IsTalking" />

<Variable x:TypeArguments="a:Bird" Name="bird" />

</Sequence.Variables>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_8" Text="Введите наименование:" />

<Assign sap2010:WorkflowViewState.IdRef="Assign\_16">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">Name</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_9" Text="Выберите окраску перьев: 0 - черная, 1 - белая, 2 - разноцветная" />

<Assign sap2010:WorkflowViewState.IdRef="Assign\_17">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">value</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_18">

<Assign.To>

<OutArgument x:TypeArguments="x:Int32">

<mca:CSharpReference x:TypeArguments="x:Int32">choice</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:Int32">

<mca:CSharpValue x:TypeArguments="x:Int32">int.Parse(value)</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Switch x:TypeArguments="x:Int32" sap2010:WorkflowViewState.IdRef="Switch`1\_3">

<Switch.Expression>

<InArgument x:TypeArguments="x:Int32">

<mca:CSharpValue x:TypeArguments="x:Int32">choice</mca:CSharpValue>

</InArgument>

</Switch.Expression>

<Assign x:Key="0" sap2010:WorkflowViewState.IdRef="Assign\_19">

<Assign.To>

<OutArgument x:TypeArguments="a:Color">

<mca:CSharpReference x:TypeArguments="a:Color">Wings</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="a:Color">

<mca:CSharpValue x:TypeArguments="a:Color">Color.Black</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign x:Key="1" sap2010:WorkflowViewState.IdRef="Assign\_20">

<Assign.To>

<OutArgument x:TypeArguments="a:Color">

<mca:CSharpReference x:TypeArguments="a:Color">Wings</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="a:Color">

<mca:CSharpValue x:TypeArguments="a:Color">Color.White</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign x:Key="2" sap2010:WorkflowViewState.IdRef="Assign\_21">

<Assign.To>

<OutArgument x:TypeArguments="a:Color">

<mca:CSharpReference x:TypeArguments="a:Color">Wings</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="a:Color">

<mca:CSharpValue x:TypeArguments="a:Color">Color.Multicolored</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

</Switch>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_10">

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">$"{Name} - говорящая? 0 - да, 1 - нет"</mca:CSharpValue>

</InArgument>

</WriteLine>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_22">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">value</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_23">

<Assign.To>

<OutArgument x:TypeArguments="x:Boolean">

<mca:CSharpReference x:TypeArguments="x:Boolean">IsTalking</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:Boolean">

<mca:CSharpValue x:TypeArguments="x:Boolean">(value == "0") ? true : false</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<local:BirdActivity Eggs="0" sap2010:WorkflowViewState.IdRef="BirdActivity\_1">

<local:BirdActivity.IsTalking>

<InArgument x:TypeArguments="x:Boolean">

<mca:CSharpValue x:TypeArguments="x:Boolean">IsTalking</mca:CSharpValue>

</InArgument>

</local:BirdActivity.IsTalking>

<local:BirdActivity.Name>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Name</mca:CSharpValue>

</InArgument>

</local:BirdActivity.Name>

<local:BirdActivity.OutAnimal>

<OutArgument x:TypeArguments="a:Bird">

<mca:CSharpReference x:TypeArguments="a:Bird">bird</mca:CSharpReference>

</OutArgument>

</local:BirdActivity.OutAnimal>

<local:BirdActivity.Wings>

<InArgument x:TypeArguments="a:Color">

<mca:CSharpValue x:TypeArguments="a:Color">Wings</mca:CSharpValue>

</InArgument>

</local:BirdActivity.Wings>

</local:BirdActivity>

<local:XMLDataSeedActivity x:TypeArguments="a:Bird" FileData="bird.xml" sap2010:WorkflowViewState.IdRef="XMLDataSeedActivity`1\_3">

<local:XMLDataSeedActivity.Object>

<InArgument x:TypeArguments="a:Bird">

<mca:CSharpValue x:TypeArguments="a:Bird">bird</mca:CSharpValue>

</InArgument>

</local:XMLDataSeedActivity.Object>

</local:XMLDataSeedActivity>

</Sequence>

<Sequence x:Key="3" sap2010:WorkflowViewState.IdRef="Sequence\_5">

<Sequence.Variables>

<Variable x:TypeArguments="x:Int32" Name="Hoofs" />

<Variable x:TypeArguments="a:Artiodactyls" Name="Arti" />

</Sequence.Variables>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_12" Text="Введите наименование" />

<Assign sap2010:WorkflowViewState.IdRef="Assign\_25">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">Name</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_13">

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">"\n" + $"{Name} - жвачное животное? 0 - да, 1 - нет"</mca:CSharpValue>

</InArgument>

</WriteLine>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_26">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">value</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_27">

<Assign.To>

<OutArgument x:TypeArguments="x:Boolean">

<mca:CSharpReference x:TypeArguments="x:Boolean">IsRum</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:Boolean">

<mca:CSharpValue x:TypeArguments="x:Boolean">(value == "0") ? true : false</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_14">

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">$"{Name} - животное с рогами? 0 - да, 1 - нет"</mca:CSharpValue>

</InArgument>

</WriteLine>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_28">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">value</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<WriteLine sap2010:WorkflowViewState.IdRef="WriteLine\_15" Text="Введите количество копыт:" />

<Assign sap2010:WorkflowViewState.IdRef="Assign\_29">

<Assign.To>

<OutArgument x:TypeArguments="x:String">

<mca:CSharpReference x:TypeArguments="x:String">value</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Console.ReadLine()</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<Assign sap2010:WorkflowViewState.IdRef="Assign\_30">

<Assign.To>

<OutArgument x:TypeArguments="x:Int32">

<mca:CSharpReference x:TypeArguments="x:Int32">Hoofs</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:Int32">

<mca:CSharpValue x:TypeArguments="x:Int32">int.Parse(value)</mca:CSharpValue>

</InArgument>

</Assign.Value>

</Assign>

<local:ArtiActivity sap2010:WorkflowViewState.IdRef="ArtiActivity\_1" IsRum="True">

<local:ArtiActivity.Hoofs>

<InArgument x:TypeArguments="x:Int32">

<mca:CSharpValue x:TypeArguments="x:Int32">Hoofs</mca:CSharpValue>

</InArgument>

</local:ArtiActivity.Hoofs>

<local:ArtiActivity.HornAvailable>

<InArgument x:TypeArguments="x:Boolean">

<mca:CSharpValue x:TypeArguments="x:Boolean">IsHorn</mca:CSharpValue>

</InArgument>

</local:ArtiActivity.HornAvailable>

<local:ArtiActivity.Name>

<InArgument x:TypeArguments="x:String">

<mca:CSharpValue x:TypeArguments="x:String">Name</mca:CSharpValue>

</InArgument>

</local:ArtiActivity.Name>

<local:ArtiActivity.OutAnimal>

<OutArgument x:TypeArguments="a:Artiodactyls">

<mca:CSharpReference x:TypeArguments="a:Artiodactyls">Arti</mca:CSharpReference>

</OutArgument>

</local:ArtiActivity.OutAnimal>

</local:ArtiActivity>

<local:XMLDataSeedActivity x:TypeArguments="a:Artiodactyls" FileData="arti.xml" sap2010:WorkflowViewState.IdRef="XMLDataSeedActivity`1\_4">

<local:XMLDataSeedActivity.Object>

<InArgument x:TypeArguments="a:Artiodactyls">

<mca:CSharpValue x:TypeArguments="a:Artiodactyls">Arti</mca:CSharpValue>

</InArgument>

</local:XMLDataSeedActivity.Object>

</local:XMLDataSeedActivity>

</Sequence>

<Assign x:Key="4" sap2010:WorkflowViewState.IdRef="Assign\_31">

<Assign.To>

<OutArgument x:TypeArguments="x:Boolean">

<mca:CSharpReference x:TypeArguments="x:Boolean">menu</mca:CSharpReference>

</OutArgument>

</Assign.To>

<Assign.Value>

<InArgument x:TypeArguments="x:Boolean">False</InArgument>

</Assign.Value>

</Assign>

</Switch>

</Sequence>

</While>

<sads:DebugSymbol.Symbol></sads:DebugSymbol.Symbol>

</Sequence>

<sap2010:WorkflowViewState.ViewStateManager>

<sap2010:ViewStateManager>

<sap2010:ViewStateData Id="WriteLine\_11" sap:VirtualizedContainerService.HintSize="477,62" />

<sap2010:ViewStateData Id="Assign\_24" sap:VirtualizedContainerService.HintSize="477,62" />

<sap2010:ViewStateData Id="Assign\_32" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="WriteLine\_1" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_1" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="WriteLine\_2" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_2" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_3" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="WriteLine\_3" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_4" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_5" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="AnimalActivity\_1" sap:VirtualizedContainerService.HintSize="242,22" />

<sap2010:ViewStateData Id="XMLDataSeedActivity`1\_1" sap:VirtualizedContainerService.HintSize="242,22" />

<sap2010:ViewStateData Id="Sequence\_1" sap:VirtualizedContainerService.HintSize="264,1023">

<sap:WorkflowViewStateService.ViewState>

<scg:Dictionary x:TypeArguments="x:String, x:Object">

<x:Boolean x:Key="IsExpanded">True</x:Boolean>

</scg:Dictionary>

</sap:WorkflowViewStateService.ViewState>

</sap2010:ViewStateData>

<sap2010:ViewStateData Id="WriteLine\_4" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_6" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="WriteLine\_5" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_7" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_8" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_9" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_10" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_11" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Switch`1\_2" sap:VirtualizedContainerService.HintSize="244,51">

<sap:WorkflowViewStateService.ViewState>

<scg:Dictionary x:TypeArguments="x:String, x:Object">

<x:Boolean x:Key="IsExpanded">False</x:Boolean>

<x:Boolean x:Key="IsPinned">False</x:Boolean>

</scg:Dictionary>

</sap:WorkflowViewStateService.ViewState>

</sap2010:ViewStateData>

<sap2010:ViewStateData Id="WriteLine\_6" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_12" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_13" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="WriteLine\_7" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_14" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_15" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="MammalActivity\_1" sap:VirtualizedContainerService.HintSize="244,22" />

<sap2010:ViewStateData Id="XMLDataSeedActivity`1\_2" sap:VirtualizedContainerService.HintSize="244,22" />

<sap2010:ViewStateData Id="Sequence\_3" sap:VirtualizedContainerService.HintSize="266,1420">

<sap:WorkflowViewStateService.ViewState>

<scg:Dictionary x:TypeArguments="x:String, x:Object">

<x:Boolean x:Key="IsExpanded">True</x:Boolean>

</scg:Dictionary>

</sap:WorkflowViewStateService.ViewState>

</sap2010:ViewStateData>

<sap2010:ViewStateData Id="WriteLine\_8" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_16" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="WriteLine\_9" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_17" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_18" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_19" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_20" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_21" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Switch`1\_3" sap:VirtualizedContainerService.HintSize="244,51">

<sap:WorkflowViewStateService.ViewState>

<scg:Dictionary x:TypeArguments="x:String, x:Object">

<x:Boolean x:Key="IsExpanded">False</x:Boolean>

<x:Boolean x:Key="IsPinned">False</x:Boolean>

</scg:Dictionary>

</sap:WorkflowViewStateService.ViewState>

</sap2010:ViewStateData>

<sap2010:ViewStateData Id="WriteLine\_10" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_22" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="Assign\_23" sap:VirtualizedContainerService.HintSize="244,62" />

<sap2010:ViewStateData Id="BirdActivity\_1" sap:VirtualizedContainerService.HintSize="244,22" />

<sap2010:ViewStateData Id="XMLDataSeedActivity`1\_3" sap:VirtualizedContainerService.HintSize="244,22" />

<sap2010:ViewStateData Id="Sequence\_4" sap:VirtualizedContainerService.HintSize="266,1114">

<sap:WorkflowViewStateService.ViewState>

<scg:Dictionary x:TypeArguments="x:String, x:Object">

<x:Boolean x:Key="IsExpanded">True</x:Boolean>

</scg:Dictionary>

</sap:WorkflowViewStateService.ViewState>

</sap2010:ViewStateData>

<sap2010:ViewStateData Id="WriteLine\_12" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_25" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="WriteLine\_13" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_26" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_27" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="WriteLine\_14" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_28" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="WriteLine\_15" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_29" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Assign\_30" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="ArtiActivity\_1" sap:VirtualizedContainerService.HintSize="242,22" />

<sap2010:ViewStateData Id="XMLDataSeedActivity`1\_4" sap:VirtualizedContainerService.HintSize="242,22" />

<sap2010:ViewStateData Id="Sequence\_5" sap:VirtualizedContainerService.HintSize="264,1227">

<sap:WorkflowViewStateService.ViewState>

<scg:Dictionary x:TypeArguments="x:String, x:Object">

<x:Boolean x:Key="IsExpanded">True</x:Boolean>

</scg:Dictionary>

</sap:WorkflowViewStateService.ViewState>

</sap2010:ViewStateData>

<sap2010:ViewStateData Id="Assign\_31" sap:VirtualizedContainerService.HintSize="242,62" />

<sap2010:ViewStateData Id="Switch`1\_1" sap:VirtualizedContainerService.HintSize="477,228" />

<sap2010:ViewStateData Id="Sequence\_6" sap:VirtualizedContainerService.HintSize="499,556">

<sap:WorkflowViewStateService.ViewState>

<scg:Dictionary x:TypeArguments="x:String, x:Object">

<x:Boolean x:Key="IsExpanded">True</x:Boolean>

</scg:Dictionary>

</sap:WorkflowViewStateService.ViewState>

</sap2010:ViewStateData>

<sap2010:ViewStateData Id="While\_1" sap:VirtualizedContainerService.HintSize="525,716" />

<sap2010:ViewStateData Id="Sequence\_2" sap:VirtualizedContainerService.HintSize="547,840">

<sap:WorkflowViewStateService.ViewState>

<scg:Dictionary x:TypeArguments="x:String, x:Object">

<x:Boolean x:Key="IsExpanded">True</x:Boolean>

</scg:Dictionary>

</sap:WorkflowViewStateService.ViewState>

</sap2010:ViewStateData>

<sap2010:ViewStateData Id="WorkflowConsoleAnimalsApp.Workflow1\_1" sap:VirtualizedContainerService.HintSize="587,920">

<sap:WorkflowViewStateService.ViewState>

<scg:Dictionary x:TypeArguments="x:String, x:Object">

<x:Boolean x:Key="ShouldExpandAll">False</x:Boolean>

</scg:Dictionary>

</sap:WorkflowViewStateService.ViewState>

</sap2010:ViewStateData>

</sap2010:ViewStateManager>

</sap2010:WorkflowViewState.ViewStateManager>

</Activity>

**AnimalActivity.cs:**

using System.Activities;

using AnimalsEntity;

namespace WorkflowConsoleAnimalsApp

{

public sealed class AnimalActivity : CodeActivity

{

public InArgument<double> Weight { get; set; }

public InArgument<string> Name { get; set; }

public InArgument<int> Age { get; set; }

public OutArgument<Animal> OutAnimal { get; set; }

// Если действие возвращает значение, создайте класс, производный от CodeActivity<TResult>

// и верните значение из метода Execute.

protected override void Execute(CodeActivityContext context)

{

// Получите значение входного аргумента Text во время выполнения

string name = context.GetValue(this.Name);

double weight = context.GetValue(this.Weight);

int age = context.GetValue(this.Age);

Animal animal = new Animal()

{

Age = Age.Get(context),

Name = Name.Get(context),

Weight = Weight.Get(context)

};

OutAnimal.Set(context, animal);

}

}

}

**MammalActivity.cs:**

using System.Activities;

using AnimalsEntity;

namespace WorkflowConsoleAnimalsApp

{

public sealed class MammalActivity : CodeActivity

{

public InArgument<TypeCover> Cover { get; set; }

public InArgument<double> Temperature { get; set; }

public InArgument<bool> IsSwimming { get; set; }

public InArgument<string> Name { get; set; }

public OutArgument<Mammal> OutAnimal { get; set; }

// Если действие возвращает значение, создайте класс, производный от CodeActivity<TResult>

// и верните значение из метода Execute.

protected override void Execute(CodeActivityContext context)

{

// Получите значение входного аргумента Text во время выполнения

TypeCover type = context.GetValue(this.Cover);

double temp = context.GetValue(this.Temperature);

bool is\_swim = context.GetValue(this.IsSwimming);

Mammal animal = new Mammal()

{

Cover = Cover.Get(context),

Temperature = Temperature.Get(context),

IsSwimming = IsSwimming.Get(context),

Name = Name.Get(context)

};

OutAnimal.Set(context, animal);

}

}

}

**ArtiActivity.cs:**

using System.Activities;

using AnimalsEntity;

namespace WorkflowConsoleAnimalsApp

{

public sealed class ArtiActivity : CodeActivity

{

public InArgument<bool> IsRum { get; set; }

public InArgument<int> Hoofs { get; set; }

public InArgument<bool> HornAvailable { get; set; }

public InArgument<string> Name { get; set; }

public OutArgument<Artiodactyls> OutAnimal { get; set; }

// Если действие возвращает значение, создайте класс, производный от CodeActivity<TResult>

// и верните значение из метода Execute.

protected override void Execute(CodeActivityContext context)

{

Artiodactyls animal = new Artiodactyls()

{

IsRum = IsRum.Get(context),

Hoofs = Hoofs.Get(context),

HornAvailable = HornAvailable.Get(context),

Name = Name.Get(context)

};

OutAnimal.Set(context, animal);

}

}

}

**BirdActivity.cs:**

using System.Activities;

using AnimalsEntity;

namespace WorkflowConsoleAnimalsApp

{

public sealed class BirdActivity : CodeActivity

{

public InArgument<bool> IsTalking { get; set; }

public InArgument<Color> Wings { get; set; }

public InArgument<int> Eggs { get; set; }

public InArgument<string> Name { get; set; }

public OutArgument<Bird> OutAnimal { get; set; }

// Если действие возвращает значение, создайте класс, производный от CodeActivity<TResult>

// и верните значение из метода Execute.

protected override void Execute(CodeActivityContext context)

{

Bird animal = new Bird()

{

IsTalking = IsTalking.Get(context),

Wings = Wings.Get(context),

Eggs = Eggs.Get(context),

Name = Name.Get(context)

};

OutAnimal.Set(context, animal);

}

}

}

**XMLDataSeedActivity.cs:**

using System;

using System.Activities;

using AnimalsEntity;

using System.Xml;

using System.Threading.Tasks;

using System.Xml.Serialization;

namespace WorkflowConsoleAnimalsApp

{

[Serializable]

public sealed class XMLDataSeedActivity<T> : CodeActivity

{

// Определите входной аргумент действия типа string

public InArgument<T> Object { get; set; }

public InArgument<string> FileData { get; set; }

// Если действие возвращает значение, создайте класс, производный от CodeActivity<TResult>

// и верните значение из метода Execute.

protected override void Execute(CodeActivityContext context)

{

// Получите значение входного аргумента Text во время выполнения

var animal = context.GetValue(this.Object);

string file\_data = context.GetValue(this.FileData);

XmlSerialize(animal, file\_data);

}

public async void XmlSerialize(T animal, string file\_path)

{

XmlWriter writer = new XmlTextWriter(file\_path, System.Text.Encoding.UTF8);

await Task.Run(() =>

{

XmlSerializer serializer = new XmlSerializer(typeof(Animal), new[]{

typeof(Mammal),

typeof(Artiodactyls), typeof(Bird) });

serializer.Serialize(writer, animal);

});

writer.Close();

}

}

}

# **Лабораторная работа №7**

Вариант 13

*Задание*: Реализовать функционал лабораторной работы № 2 с использованием технологии WPF (Windows Presentation Foundation) и плагина Fody для привязки данных (либо другой технологии привязки данных).

*Результат выполнения программы:*

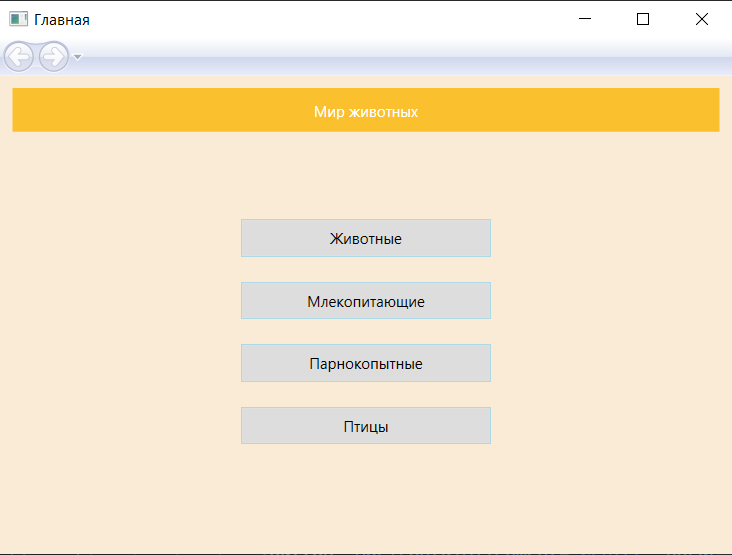


Рис. 43 – Главная страница

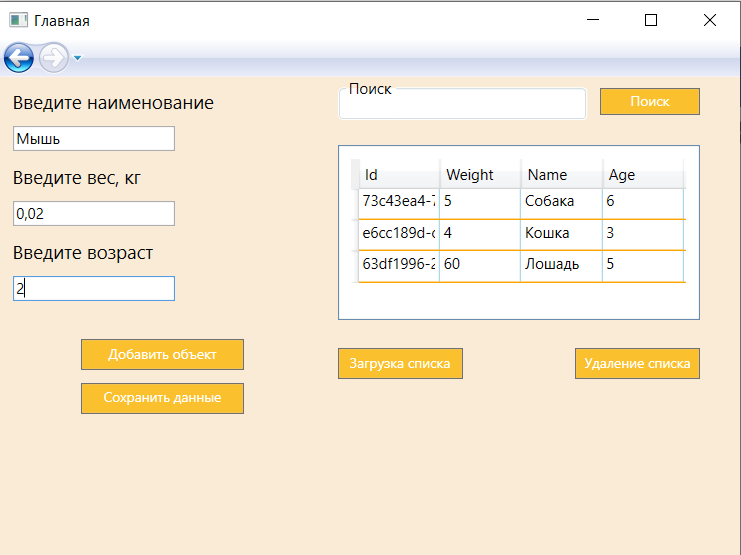


Рис. 44 – Форма заполнения сведений о животных

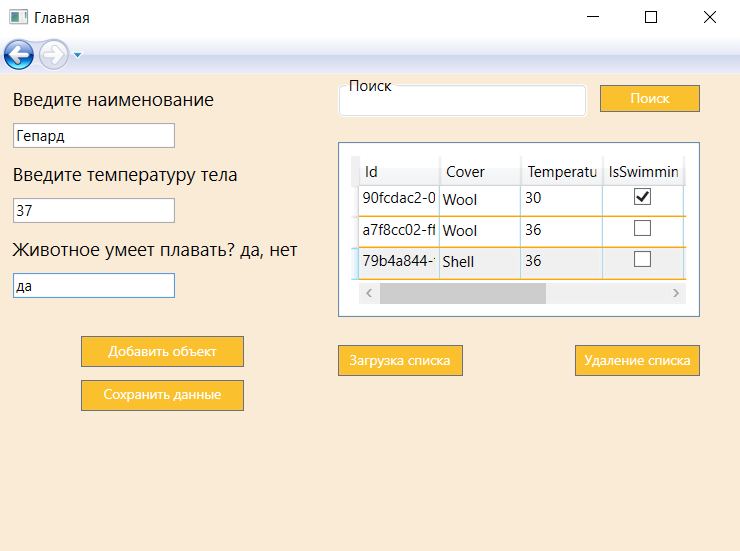


Рис. 45 – Форма заполнения сведений о млекопитающих

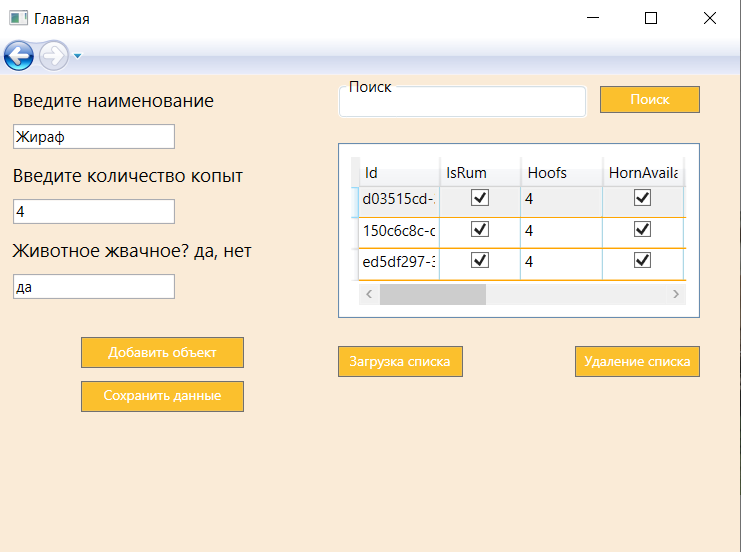


Рис. 46 – Форма заполнения сведений о парнокопытных

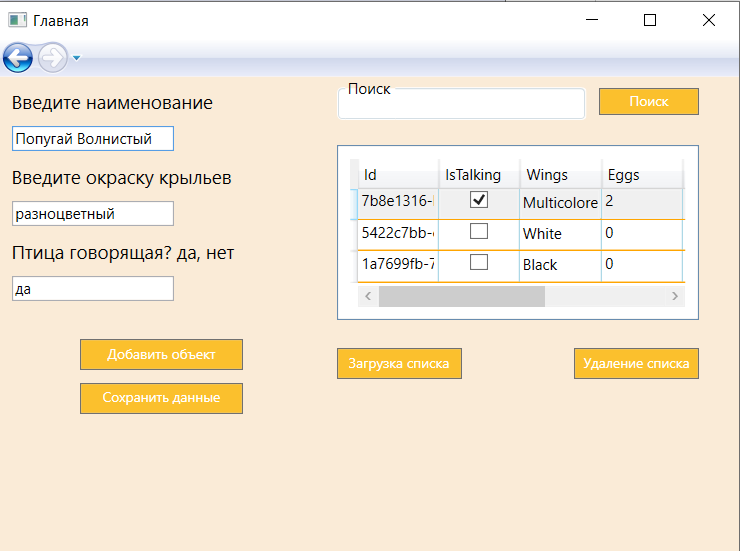


Рис. 47 – Форма заполнения сведений о птицах

*Листинг программных модулей:*

\*AnimalsEntity - тот же, что и во второй лабораторной работе.

**HomePage.xaml:**

<Page x:Class="WpfAnimalsProject.HomePage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:WpfAnimalsProject"

mc:Ignorable="d"

Background="AntiqueWhite"

d:DesignHeight="380" d:DesignWidth="530"

Title="Главная">

<Grid>

<Grid Margin="10,10,10,10">

<StackPanel Orientation="Vertical">

<Border Height="35" Padding="5" Background="#FBC02D">

<Label HorizontalAlignment="Center" VerticalAlignment="Center" Foreground="White">Мир животных</Label>

</Border>

<Button HorizontalAlignment="Center" BorderBrush="LightBlue" FontSize="12" Width="200" Height="30" Margin="0,70,0,0" Click="Button\_Animal">Животные</Button>

<Button HorizontalAlignment="Center" BorderBrush="LightBlue" Height="30" Width="200" FontSize="12" Margin="0,20,0,0" Click="Button\_Mammal">Млекопитающие</Button>

<Button HorizontalAlignment="Center" BorderBrush="LightBlue" Height="30" Width="200" FontSize="12" Margin="0,20,0,0" Click="Button\_Anti">Парнокопытные</Button>

<Button HorizontalAlignment="Center" BorderBrush="LightBlue" Height="30" Width="200" FontSize="12" Margin="0,20,0,0" Click="Button\_Bird">Птицы</Button>

</StackPanel>

</Grid>

</Grid>

</Page>

**HomePage.xaml.cs :**

using System.Windows;

using System.Windows.Controls;

namespace WpfAnimalsProject

{

/// <summary>

/// Логика взаимодействия для HomePage.xaml

/// </summary>

public partial class HomePage : Page

{

public enum AnimalType

{

Animal, Mammal, Anti, Bird

}

public HomePage()

{

InitializeComponent();

}

private void Button\_Animal(object sender, RoutedEventArgs e)

{

// View Page

StaticAnimal.TypeOfAnimal = AnimalType.Animal;

AnimalPage expenseReportPage = new AnimalPage();

this.NavigationService.Navigate(expenseReportPage);

}

private void Button\_Mammal(object sender, RoutedEventArgs e)

{

// View Page

StaticAnimal.TypeOfAnimal = AnimalType.Mammal;

AnimalPage expenseReportPage = new AnimalPage();

this.NavigationService.Navigate(expenseReportPage);

}

private void Button\_Anti(object sender, RoutedEventArgs e)

{

// View Page

StaticAnimal.TypeOfAnimal = AnimalType.Anti;

AnimalPage expenseReportPage = new AnimalPage();

this.NavigationService.Navigate(expenseReportPage);

}

private void Button\_Bird(object sender, RoutedEventArgs e)

{

// View Page

StaticAnimal.TypeOfAnimal = AnimalType.Bird;

AnimalPage expenseReportPage = new AnimalPage();

this.NavigationService.Navigate(expenseReportPage);

}

}

}

**AnimalPage.xaml:**

<Page x:Class="WpfAnimalsProject.AnimalPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:WpfAnimalsProject"

mc:Ignorable="d"

Background="AntiqueWhite"

d:DesignHeight="300" d:DesignWidth="580"

Title="AnimalPage">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="260"/>

<ColumnDefinition Width="auto"/>

</Grid.ColumnDefinitions>

<StackPanel Orientation="Vertical" Margin="10" Grid.Column="0">

<TextBlock x:Name="txt\_first" FontSize="15" Text="{Binding ElementName=txt\_first,Path=Text}"></TextBlock>

<TextBox Name="edit\_first" Width="130" HorizontalAlignment="Left" Height="20" Margin="0,10,0,0"></TextBox>

<TextBlock x:Name="txt\_second" Text="{Binding ElementName=txt\_second,Path=Text}" Margin="0,10,0,0" FontSize="15"></TextBlock>

<TextBox Name="edit\_second" Width="130" HorizontalAlignment="Left" Height="20" Margin="0,10,0,0"></TextBox>

<TextBlock x:Name="txt\_third" Text="{Binding ElementName=txt\_third,Path=Text}" Margin="0,10,0,0" FontSize="15"></TextBlock>

<TextBox Name="edit\_third" Width="130" HorizontalAlignment="Left" Height="20" Margin="0,10,0,0"></TextBox>

<Button Width="130" FontFamily="bold" FontWeight="Regular"

Padding="3" Height="25" Background="#FBC02D" Click="Button\_Add"

Foreground="White" Margin="0,30,0,0" FontSize="11">

Добавить объект

</Button>

<Button Width="130" FontFamily="bold" FontWeight="Regular"

Padding="3" Height="25" Background="#FBC02D" Click="Button\_Save"

Foreground="White" Margin="10" FontSize="11">

Сохранить данные

</Button>

</StackPanel>

<StackPanel Orientation="Vertical" Grid.Column="1">

<StackPanel Orientation="Horizontal">

<GroupBox Background="White" HorizontalAlignment="Left" Width="200" Header="Поиск" Margin="10,0,0,0" Height="35">

<TextBox Name="edit\_search" Foreground="Green" BorderBrush="White" Height="35" FontSize="10">

</TextBox>

</GroupBox>

<Button FontFamily="bold" FontWeight="Regular"

Margin="10,5,0,0" Width="80" Click="Button\_Search"

Padding="3" Height="22" Background="#FBC02D"

Foreground="White" FontSize="11">

Поиск

</Button>

</StackPanel>

<DataGrid BorderThickness="1"

Padding="10" Margin="10,20,0,0"

ColumnWidth="65" HeadersVisibility="All"

GridLinesVisibility="All" HorizontalGridLinesBrush="Orange"

VerticalGridLinesBrush="LightBlue" AutoGenerateColumns="true"

Height="140" VerticalAlignment="Top" Background="White"

AreRowDetailsFrozen="True" Width="290"

SelectionMode="Single" RowHeight="25" CanUserAddRows="False"

CanUserDeleteRows="False" ItemsSource="{Binding}"

Name="tb\_list" CanUserSortColumns = "False">

</DataGrid>

<StackPanel Orientation="Horizontal" Margin="0,17,0,0" HorizontalAlignment="Left">

<Button FontFamily="bold" FontWeight="Regular"

Margin="10,5,0,0" Width="100" Click="Button\_Load"

Padding="3" Height="25" Background="#FBC02D"

Foreground="White" FontSize="11">

Загрузка списка

</Button>

<Button FontFamily="bold" FontWeight="Regular"

Margin="90,5,0,0" Width="100" Click="Button\_Delete"

Padding="3" Height="25" Background="#FBC02D"

Foreground="White" FontSize="11">

Удаление списка

</Button>

</StackPanel>

</StackPanel>

</Grid>

</Page>

**AnimalPage.xaml.cs:**

using AnimalsEntity;

using System.Collections.Generic;

using System.ComponentModel;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Navigation;

using static WpfAnimalsProject.HomePage;

namespace WpfAnimalsProject

{

/// <summary>

/// Логика взаимодействия для AnimalPage.xaml

/// </summary>

public partial class AnimalPage : Page

{

AnimalType type;

public List<Animal> Animals = new List<Animal>();

public List<Artiodactyls> Artiodactyls = new List<Artiodactyls>();

public List<Mammal> Mammals = new List<Mammal>();

public List<Bird> Birds = new List<Bird>();

FileData<Animal> animal\_data;

FileData<Bird> bird\_data;

FileData<Artiodactyls> arti\_data;

FileData<Mammal> mammal\_data;

public AnimalPage()

{

InitializeComponent();

var parentWindow = this.Parent as Window;

NavigationService nav = NavigationService.GetNavigationService(this);

//if (parentWindow != null)

// parentWindow.Loaded += ParentWindow\_Loaded;

//this.NavigationService.LoadCompleted += NavigationService\_LoadCompleted;

type = StaticAnimal.TypeOfAnimal;

Binding binding = new Binding();

Binding binding\_2 = new Binding();

Binding binding\_3 = new Binding();

binding.ElementName = "txt\_first"; // элемент-источник

binding.Path = new PropertyPath("Text"); // свойство элемента-источника

txt\_first.SetBinding(TextBlock.TextProperty, binding); // установка привязки для элемента-приемника

binding\_2.ElementName = "txt\_second"; // элемент-источник

binding\_2.Path = new PropertyPath("Text"); // свойство элемента-источника

txt\_second.SetBinding(TextBlock.TextProperty, binding\_2); // установка привязки для элемента-приемника

binding\_3.ElementName = "txt\_third"; // элемент-источник

binding\_3.Path = new PropertyPath("Text"); // свойство элемента-источника

txt\_third.SetBinding(TextBlock.TextProperty, binding\_3); // установка привязки для элемента-приемника

switch (type)

{

case AnimalType.Animal:

{

animal\_data = new FileData<Animal>(Animals, "animals.xml");

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите вес, кг";

txt\_third.Text = "Введите возраст";

}

break;

case AnimalType.Mammal:

{

mammal\_data = new FileData<Mammal>(Mammals, "mammals.xml");

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите температуру тела";

txt\_third.Text = "Животное умеет плавать? да, нет";

}

break;

case AnimalType.Anti:

{

arti\_data = new FileData<Artiodactyls>(Artiodactyls, "arti.xml");

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите количество копыт";

txt\_third.Text = "Животное жвачное? да, нет";

}

break;

case AnimalType.Bird:

{

bird\_data = new FileData<Bird>(Birds, "birds.xml");

txt\_first.Text = "Введите наименование";

txt\_second.Text = "Введите окраску крыльев";

txt\_third.Text = "Птица говорящая? да, нет";

}

break;

default:

break;

}

}

private void Button\_Add(object sender, RoutedEventArgs e)

{

AddInfo();

}

private async void AddInfo()

{

if (string.IsNullOrWhiteSpace(txt\_first.Text) || string.IsNullOrWhiteSpace(txt\_second.Text)

|| string.IsNullOrWhiteSpace(txt\_third.Text))

{

MessageBox.Show("Введите все данные");

return;

}

switch (type)

{

case AnimalType.Animal:

{

Animals = await animal\_data.Deserialize();

Animal animal = new Animal();

animal.Age = int.Parse(edit\_third.Text);

animal.Name = edit\_first.Text;

animal.Weight = int.Parse(edit\_second.Text);

Animals.Add(animal);

animal\_data.objects = Animals;

animal\_data.XmlSerialize();

tb\_list.DataContext = Animals.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

case AnimalType.Mammal:

{

Mammals = await mammal\_data.Deserialize();

Mammal animal = new Mammal();

animal.Temperature = int.Parse(edit\_second.Text);

animal.Name = edit\_first.Text;

animal.IsSwimming = (edit\_third.Text == "да")? true: false;

Mammals.Add(animal);

mammal\_data.objects = Mammals;

mammal\_data.XmlSerialize();

tb\_list.DataContext = Mammals.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

case AnimalType.Anti:

{

Artiodactyls = await arti\_data.Deserialize();

Artiodactyls animal = new Artiodactyls();

animal.IsRum = (edit\_third.Text == "да") ? true : false;

animal.Name = edit\_first.Text;

animal.Hoofs= int.Parse(edit\_second.Text);

Artiodactyls.Add(animal);

arti\_data.objects = Artiodactyls;

arti\_data.XmlSerialize();

tb\_list.DataContext = Artiodactyls.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

case AnimalType.Bird:

{

Birds = await bird\_data.Deserialize();

Bird animal = new Bird();

animal.Wings = (edit\_second.Text == "черный") ? Color.Black : (edit\_second.Text == "белый") ? Color.White :

(edit\_second.Text == "разноцветный") ? Color.Multicolored : Color.Black;

animal.Name = edit\_first.Text;

animal.IsTalking = (edit\_third.Text == "да") ? true : false;

Birds.Add(animal);

bird\_data.objects = Birds;

bird\_data.XmlSerialize();

tb\_list.DataContext = Birds.ToList();

edit\_first.Text = "";

edit\_second.Text = "";

edit\_third.Text = "";

}

break;

default:

break;

}

}

private void Button\_Save(object sender, RoutedEventArgs e)

{

switch (type)

{

case AnimalType.Animal:

{

Animals = tb\_list.Items.OfType<Animal>().ToList();

animal\_data.objects = Animals;

animal\_data.XmlSerialize();

}

break;

case AnimalType.Mammal:

{

Mammals = tb\_list.Items.OfType<Mammal>().ToList();

mammal\_data.objects = Mammals;

mammal\_data.XmlSerialize();

}

break;

case AnimalType.Anti:

{

Artiodactyls = tb\_list.Items.OfType<Artiodactyls>().ToList();

arti\_data.objects = Artiodactyls;

arti\_data.XmlSerialize();

}

break;

case AnimalType.Bird:

{

Birds = tb\_list.Items.OfType<Bird>().ToList();

bird\_data.objects = Birds;

bird\_data.XmlSerialize();

}

break;

default:

break;

}

}

private void Button\_Load(object sender, RoutedEventArgs e)

{

LoadInfo();

}

private async void LoadInfo()

{

switch (type)

{

case AnimalType.Animal:

{

Animals = await animal\_data.Deserialize();

tb\_list.DataContext = Animals.ToList();

}

break;

case AnimalType.Mammal:

{

Mammals = await mammal\_data.Deserialize();

tb\_list.DataContext = Mammals.ToList();

}

break;

case AnimalType.Anti:

{

Artiodactyls = await arti\_data.Deserialize();

tb\_list.DataContext = Artiodactyls.ToList();

}

break;

case AnimalType.Bird:

{

Birds = await bird\_data.Deserialize();

tb\_list.DataContext = Birds.ToList();

}

break;

default:

break;

}

}

private void Button\_Search(object sender, RoutedEventArgs e)

{

switch (type)

{

case AnimalType.Animal:

{

BindingList<Animal> BS = new BindingList<Animal>(Animals.Where(m => m.Age.ToString().Contains(edit\_search.Text) || m.Name.ToLower().Contains(edit\_search.Text.ToLower()) || m.Weight.ToString().Contains(edit\_search.Text)).ToList());

tb\_list.DataContext = BS;

}

break;

case AnimalType.Mammal:

{

BindingList<Mammal> BS = new BindingList<Mammal>(Mammals.Where(m => m.Name.ToLower().Contains(edit\_search.Text.ToLower())).ToList());

tb\_list.DataContext = BS;

}

break;

case AnimalType.Anti:

{

BindingList<Artiodactyls> BS = new BindingList<Artiodactyls>(Artiodactyls.Where(m => m.Name.ToLower().Contains(edit\_search.Text.ToLower())).ToList());

tb\_list.DataContext = BS;

}

break;

case AnimalType.Bird:

{

BindingList<Bird> BS = new BindingList<Bird>(Birds.Where(m => m.Name.ToLower().Contains(edit\_search.Text.ToLower())).ToList());

tb\_list.DataContext = BS;

}

break;

default:

break;

}

}

private void Button\_Delete(object sender, RoutedEventArgs e)

{

switch (type)

{

case AnimalType.Animal:

{

Animals = new List<Animal>();

animal\_data.objects = Animals;

animal\_data.XmlSerialize();

tb\_list.DataContext = Animals.ToList();

}

break;

case AnimalType.Mammal:

{

Mammals = new List<Mammal>();

mammal\_data.objects = Mammals;

mammal\_data.XmlSerialize();

tb\_list.DataContext = Mammals.ToList();

}

break;

case AnimalType.Anti:

{

Artiodactyls = new List<Artiodactyls>();

arti\_data.objects = Artiodactyls;

arti\_data.XmlSerialize();

tb\_list.DataContext = Artiodactyls.ToList();

}

break;

case AnimalType.Bird:

{

Birds = new List<Bird>();

bird\_data.objects = Birds;

bird\_data.XmlSerialize();

tb\_list.DataContext = Birds.ToList();

}

break;

default:

break;

}

}

}

}