Лабораторная работа №12+13

Экономическая информационная система.

Отчёт.

Автор: Шамаев Григорий

Григорьевич

Дата сдачи: 10.02.2015

I. ***Цель работы:***

Разработка экономической информационной системы с использованием XML на языке программирования С# посредствам MS VisualStudio 2012.

II. ***Задание:***

Разработайте приложение с использованием элементов Label, Button, TextBox, ComboBox, Panel, GroupBox, LinkLabel, ErrorProvider, RadioButton, CheckBox.Приложение должно позволять добавлять, удалять и редактировать информацию. Информация должна храниться в одном или нескольких файлах. Переключение между объектами проводить кнопками «Вперед» и «Назад» с указанием текущего номера объекта (или с помощью TrackBar). Управление основными объектами реализовать с помощью классов или структур. Добавьте также элементы календаря и выбора даты / времени. Ввод неправильной информации ограничить с помощью обработки исключений и элемента ErrorProvider.

Управление заявками химчистки-прачечной. С указанием клиента, типа операции, цены, времени операции, номенклатуры вещей.

III. ***Описание полей в форме:***

groupBox1 - Рамка для главных полей с номером заявки;

label1/2/3/4/5 - названия полей;

ClientBox - Ф.И.О. клиента;

HimButton,ObStButton - выбор типа операции;

TimeBox - время операции в минутах;

CostBox - стоимость операции;

NomenBox - номенклатура вещей;

DateTimeBox - дата создания заявки;

Link - переход на указанный URL адрес;

***Кнопки:***

BackButton - переход к следущей заявке;

NextButton - переход к предыдущей заявке;

AddButton - создание новой заявки;

DelButton - удаление текущей заявки;

EditButton - редактирование текущей заявки;

CloseButton - закрытие формы;

IV.***Текст программы:***

**Класс главной формы "Form2.cs":**

using System;

using System.Xml;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Lab12\_Var19

{

public partial class Form2 : Form

{

Form main;

int tek;

Zayavki zayavki;

Opers opers;

Clients clients;

public Form2(Form main)

{

this.main = main;

InitializeComponent();

opers = new Opers();

clients = new Clients();

zayavki = new Zayavki(clients, opers);

tek = zayavki.getMax();

Zayavka Z = zayavki.findByID(tek);

FillComboBoxes();

this.ShowData(Z);

this.Check();

}

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

{

main.Show();

Close();

}

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

{

tek = zayavki.getMax() + 1;

groupBox1.Text = "Заявка №" + tek;

ClientBox.Text = "";

CostBox.Text = "";

TimeBox.Text = "";

DateTimeBox.Value = DateTime.Today;

DelButton.Enabled = true;

Check();

}

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

{

if (tek >= zayavki.getMax())

{

Client a = clients.findByFIO(ClientBox.Text);

if (a == null)

{

Client C = new Client();

C.FIO = ClientBox.Text;

C.ID = clients.getMax() + 1;

clients.AddClient(C);

clients.SaveClients();

}

}

int O;

if (HimButton.Checked == true)

{

O = 1;

}

else

{

O = 2;

}

Zayavka Z = new Zayavka(tek, ClientBox.Text, O, CostBox.Text, TimeBox.Text, NomenBox.Text, Convert.ToString(DateTimeBox.Value), clients, opers);

if (Z.ID <= zayavki.getMax())

{

zayavki.findByID(Z.ID).client = Z.client;

zayavki.findByID(Z.ID).oper = Z.oper;

zayavki.findByID(Z.ID).cost = Z.cost;

zayavki.findByID(Z.ID).time = Z.time;

zayavki.findByID(Z.ID).nomen = Z.nomen;

zayavki.findByID(Z.ID).date = Z.date;

}

else

{

zayavki.AddZayavka(Z);

}

zayavki.SaveZayavki();

FillComboBoxes();

Check();

}

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

{

Zayavka Z = zayavki.Prev(tek);

this.ShowData(Z);

}

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

{

Zayavka Z = zayavki.Next(tek);

ShowData(Z);

}

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

{

bool a = true;

Zayavka Z = zayavki.findByID(tek);

Zayavka next = zayavki.Next(tek);

if (next.ID == tek)

{

next = zayavki.Prev(tek);

if (next.ID == tek)

{

tek = 1;

groupBox1.Text = "Заявка №" + 1;

ClientBox.Text = "";

CostBox.Text = "";

TimeBox.Text = "";

DateTimeBox.Value = DateTime.Today;

Check();

a = false;

DelButton.Enabled = false;

}

}

zayavki.DelZayavka(Z);

zayavki.SaveZayavki();

if (a)

ShowData(next);

}

private void HimButton\_CheckedChanged(object sender, EventArgs e)

{

CalculateCost();

}

private void ObStButton\_CheckedChanged(object sender, EventArgs e)

{

CalculateCost();

}

private void ShowData(Zayavka Z)

{

tek = Z.ID;

groupBox1.Text = "Заявка №" + Z.ID;

ClientBox.Text = Z.client.FIO;

TimeBox.Text = Z.time;

int cost = 0;

if (Z.oper.ID == 1)

{

cost = 20;

}

else

{

cost = 10;

}

cost = cost \* Convert.ToInt32(Z.time);

if (Z.oper.ID == 1)

{

HimButton.Checked = true;

ObStButton.Checked = false;

}

else

{

HimButton.Checked = false;

ObStButton.Checked = true;

}

CostBox.Text = Convert.ToString(cost);

NomenBox.Text = Z.nomen;

DateTimeBox.Value = Convert.ToDateTime(Z.date);

Check();

}

private void Check()

{

if (tek >= zayavki.getMax())

{

NextButton.Enabled = false;

}

else

{

NextButton.Enabled = true;

}

if (tek == zayavki.getMin())

{

BackButton.Enabled = false;

}

else

{

BackButton.Enabled = true;

}

}

private void FillComboBoxes()

{

ClientBox.Items.Clear();

foreach (Client C in clients.getList())

{

ClientBox.Items.Add(C.FIO);

}

}

private void Link\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

System.Diagnostics.Process.Start("http://troll-face.ru/static/a/f/3b/zdorovopravda-B6gPaM.jpg");

}

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

{

try

{

Convert.ToInt32(TimeBox.Text);

errorProvider1.Clear();

CalculateCost();

}

catch (FormatException)

{

errorProvider1.SetError(TimeBox, "Нужно ввести число");

}

}

private void CalculateCost()

{

int cost = 0;

if (HimButton.Checked == true)

{

cost = 20;

}

else

{

cost = 10;

}

cost = cost \* Convert.ToInt32(TimeBox.Text);

CostBox.Text = Convert.ToString(cost);

}

private void ClientBox\_TextUpdate(object sender, EventArgs e)

{

int a = 0;

int b = 0;

char[] str = ClientBox.Text.ToCharArray();

foreach (char s in str)

{

a++;

try

{

int c = Convert.ToInt32(Convert.ToString(s));

}

catch (FormatException)

{

b++;

}

}

if (a==b)

errorProvider1.Clear();

else

errorProvider1.SetError(ClientBox, "Поле не может содержать цифры");

}

}

}

**Класс заявок "Zayavka.cs":**

using System;

using System.Xml;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab12\_Var19

{

class Zayavka

{

public int ID;

public Client client;

public Oper oper;

public int cost;

public string time;

public string nomen;

public string date;

public Zayavka(){}

public Zayavka(int tek,string C, int O, string Cost, string Time, string Nomen,string date,Clients clients, Opers opers )

{

this.ID = tek;

this.client = clients.findByFIO(C);

this.oper = opers.findByID(O);

this.cost = Convert.ToInt32(Cost);

this.time = Time;

this.nomen = Nomen;

this.date = date;

}

}

class Zayavki

{

List<Zayavka> AllZayavki;

public Zayavki(Clients clients, Opers opers)

{

AllZayavki = new List<Zayavka>();

AllZayavki = readZayavki(clients, opers);

}

public List<Zayavka> readZayavki(Clients clients, Opers opers)

{

List<Zayavka> catalog = new List<Zayavka>();

XmlDocument xDoc = new XmlDocument();

xDoc.Load("../../DataBase.xml");

XmlElement Main = xDoc.DocumentElement;// получим корневой элемент

foreach (XmlNode xRoot in Main)// обход всех узлов в корневом элементе

{

if (xRoot.Name == "Zayavki")

{

foreach (XmlNode xnode in xRoot)

{

Zayavka temp = new Zayavka();

XmlNode attr = xnode.Attributes.GetNamedItem("ID");

temp.ID = Convert.ToInt32(attr.Value);

// обходим все дочерние узлы

foreach (XmlNode childnode in xnode.ChildNodes)

{

if (childnode.Name == "Client")

{

temp.client = clients.findByID(Convert.ToInt32(childnode.InnerText));

}

if (childnode.Name == "OperrationType")

{

temp.oper = opers.findByID(Convert.ToInt32(childnode.InnerText));

}

if (childnode.Name == "Cost")

{

temp.cost = Convert.ToInt32(childnode.InnerText);

}

if (childnode.Name == "During")

{

temp.time = childnode.InnerText;

}

if (childnode.Name == "Nomenklatura")

{

temp.nomen = childnode.InnerText;

}

if (childnode.Name == "Date")

{

temp.date = childnode.InnerText;

}

}

catalog.Add(temp);

}

}

}

return catalog;

}

public Zayavka Next(int ID)

{

int index = AllZayavki.FindIndex(x => x.ID == ID);

if (index < AllZayavki.Count - 1)

{

return AllZayavki[index + 1];

}

return AllZayavki[index];

}

public Zayavka Prev(int ID)

{

int index = AllZayavki.FindIndex(x => x.ID == ID);

if (index > 0)

{

return AllZayavki[index - 1];

}

if (index < 0)

{

int index1 = AllZayavki.FindIndex(x => x.ID == ID-1);

return AllZayavki[index1];

}

return AllZayavki[0];

}

public int getMax()

{

int max = 0;

foreach (Zayavka Z in AllZayavki)

{

if (max < Z.ID) max = Z.ID;

}

return max;

}

public int getMin()

{

int min = 0;

if (AllZayavki.Count > 0)

{

min = AllZayavki[0].ID;

foreach (Zayavka Z in AllZayavki)

{

if (min > Z.ID) min = Z.ID;

}

}

return min;

}

public Zayavka findByID(int ID)

{

return AllZayavki.Find(x => x.ID == ID);

}

public void AddZayavka(Zayavka Z)

{

AllZayavki.Add(Z);

}

public void DelZayavka(Zayavka Z)

{

AllZayavki.Remove(Z);

}

public void SaveZayavki()

{

XmlDocument xDoc = new XmlDocument();

xDoc.Load("../../DataBase.xml");

XmlElement Main = xDoc.DocumentElement;

foreach (XmlElement xRoot in Main)

{

if (xRoot.Name == "Zayavki")

{

xRoot.RemoveAll();

foreach (Zayavka Z in AllZayavki)

{

XmlNode Zayavka = xDoc.CreateElement("Zayavka");

xRoot.AppendChild(Zayavka);//указываем родителя

XmlAttribute ID = xDoc.CreateAttribute("ID");//созд атрибут

ID.Value = Convert.ToString(Z.ID);//уст значение

Zayavka.Attributes.Append(ID);//присваиваем

XmlNode Client = xDoc.CreateElement("Client"); // даём имя

Client.InnerText = Convert.ToString(Z.client.ID); // и значение

Zayavka.AppendChild(Client); // и указываем кому принадлежит

XmlNode OperrationType = xDoc.CreateElement("OperrationType");

OperrationType.InnerText = Convert.ToString(Z.oper.ID);

Zayavka.AppendChild(OperrationType);

XmlNode Cost = xDoc.CreateElement("Cost");

Cost.InnerText = Convert.ToString(Z.cost);

Zayavka.AppendChild(Cost);

XmlNode During = xDoc.CreateElement("During");

During.InnerText = Z.time;

Zayavka.AppendChild(During);

XmlNode Nomen = xDoc.CreateElement("Nomenklatura");

Nomen.InnerText = Z.nomen;

Zayavka.AppendChild(Nomen);

XmlNode Date = xDoc.CreateElement("Date");

Date.InnerText = Z.date;

Zayavka.AppendChild(Date);

}

}

}

xDoc.Save("../../DataBase.xml");

}

}

}

**Класс клиентов "Client.cs":**

using System;

using System.Xml;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab12\_Var19

{

class Client

{

public int ID;

public string FIO;

public Client()

{

}

}

class Clients

{

List<Client> allClients;

public Clients()

{

allClients = new List<Client>();

allClients = readClients();

}

public List<Client> readClients()

{

List<Client> catalog = new List<Client>();

XmlDocument xDoc = new XmlDocument();

xDoc.Load("../../DataBase.xml");

XmlElement Main = xDoc.DocumentElement;// получим корневой элемент

foreach (XmlElement xRoot in Main)

{

if (xRoot.Name == "clients")

{

foreach (XmlNode xnode in xRoot)// обход всех узлов в корневом элементе

{

Client temp = new Client();

XmlNode attr = xnode.Attributes.GetNamedItem("ID");

temp.ID = Convert.ToInt32(attr.Value);

// обходим все дочерние узлы

foreach (XmlNode childnode in xnode.ChildNodes)

{

if (childnode.Name == "FIO")

{

temp.FIO = childnode.InnerText;

}

}

catalog.Add(temp);

}

}

}

return catalog;

}

public List<Client> getList()

{

return allClients;

}

public Client findByID(int ID)

{

return allClients.Find(x => x.ID == ID);

}

public Client findByFIO(string FIO)

{

return allClients.Find(x => x.FIO == FIO);

}

public void AddClient(Client C)

{

allClients.Add(C);

}

public void SaveClients()

{

XmlDocument xDoc = new XmlDocument();

xDoc.Load("../../DataBase.xml");

XmlElement Main = xDoc.DocumentElement;

foreach (XmlElement xRoot in Main)

{

if (xRoot.Name == "clients")

{

xRoot.RemoveAll();

foreach (Client C in allClients)

{

XmlNode Client = xDoc.CreateElement("Client");

xRoot.AppendChild(Client);//указываем родителя

XmlAttribute ID = xDoc.CreateAttribute("ID");//созд атрибут

ID.Value = Convert.ToString(C.ID);//уст значение

Client.Attributes.Append(ID);//присваиваем

XmlNode FIO = xDoc.CreateElement("FIO"); // даём имя

FIO.InnerText = C.FIO; // и значение

Client.AppendChild(FIO); // и указываем кому принадлежит

}

}

}

xDoc.Save("../../DataBase.xml");

}

public int getMax()

{

int max = 0;

foreach (Client C in allClients)

{

if (max < C.ID) max = C.ID;

}

return max;

}

}

}

**Класс типов операций "Oper.cs":**

using System;

using System.Xml;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab12\_Var19

{

class Oper

{

public int ID;

public string Style;

}

class Opers

{

List<Oper> allOpers;

public Opers()

{

allOpers = new List<Oper>();

allOpers = readOpers();

}

public List<Oper> readOpers()

{

List<Oper> catalog = new List<Oper>();

XmlDocument xDoc = new XmlDocument();

xDoc.Load("../../DataBase.xml");

XmlElement Main = xDoc.DocumentElement;// получим корневой элемент

foreach (XmlElement xRoot in Main)

{

foreach (XmlNode xnode in xRoot)// обход всех узлов в корневом элементе

{

Oper temp = new Oper();

XmlNode attr = xnode.Attributes.GetNamedItem("ID");

temp.ID = Convert.ToInt32(attr.Value);

// обходим все дочерние узлы

foreach (XmlNode childnode in xnode.ChildNodes)

{

if (childnode.Name == "style")

{

temp.Style = childnode.InnerText;

}

}

catalog.Add(temp);

}

}

return catalog;

}

public List<Oper> getList()

{

return allOpers;

}

public Oper findByID(int ID)

{

return allOpers.Find(x => x.ID == ID);

}

public Oper findByStyle(string Style)

{

return allOpers.Find(x => x.Style == Style);

}

}

}

V.***XML документ "DataBase.xml":***

<?xml version="1.0" encoding="utf-8"?>

<Main>

<Zayavki>

<Zayavka ID="1">

<Client>1</Client>

<OperrationType>1</OperrationType>

<Cost>2000</Cost>

<During>100</During>

<Nomenklatura>8013264</Nomenklatura>

<Date>07.01.2015 0:00:00</Date>

</Zayavka>

.........список заявок

</Zayavki>

<clients>

<Client ID="1">

<FIO>Шамаев Григорий Григорьевич</FIO>

</Client>

......//список клиентов

</clients>

<opers> // типы операций

<oper ID="1">

<style>Химчистка</style>

</oper>

<oper ID="2">

<style>Обычная стирка</style>

</oper>

</opers>

</Main>