**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ИМЕНИ ПАВЛА ОСИПОВИЧА СУХОГО**

Факультет автоматизированных информационных систем

Кафедра «Информационные технологии»

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №13

по дисциплине:

«**ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ**»

на тему:

«**Разработка игр с графическим интерфейсом**»

Выполнил: Привалов С.А.

Принял: Трубенок Д. Н.

Дата сдачи отчёта:

Дата допуска к защите:

Дата защиты:

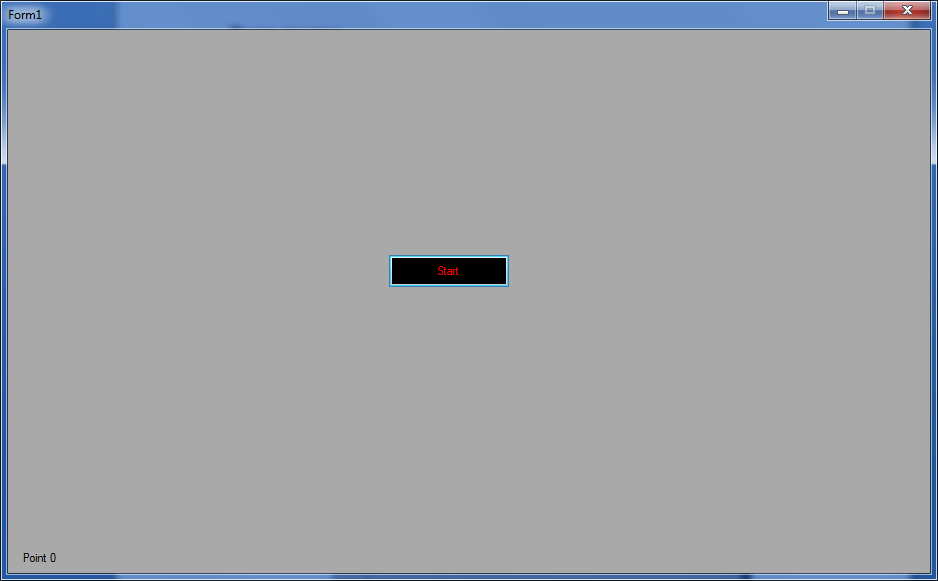
Гомель 2015

**Цель**: получить навыки разработки программ-игр, использующих формы, элементы управления и принципы событийно-управляемого программирования.

**Вариант “Змейка”**

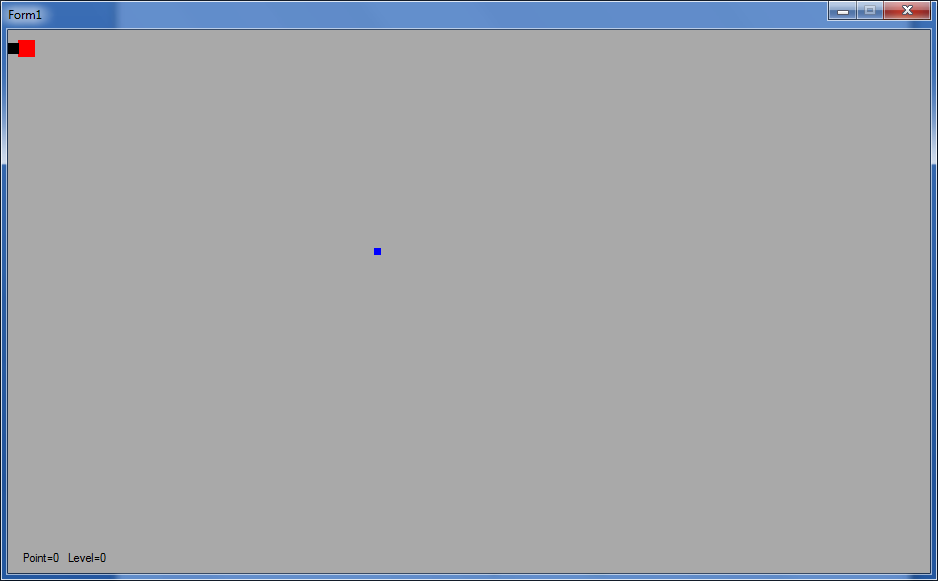
**Выполнение:**

При запуске приложения вам представится возможность начать игру отображено на картинке 1.



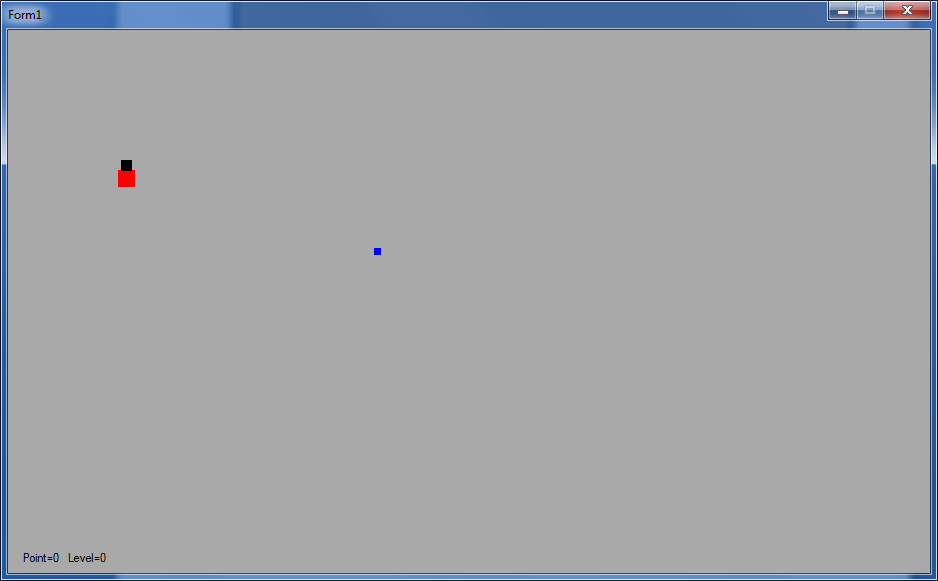
Картинка 1 – Начальная стадия игры

При нажатии на кнопку “*Start*” вам отобразится объект “Змейка”, а так же квадратик, который нужно собрать, что отображено на картинка 2.

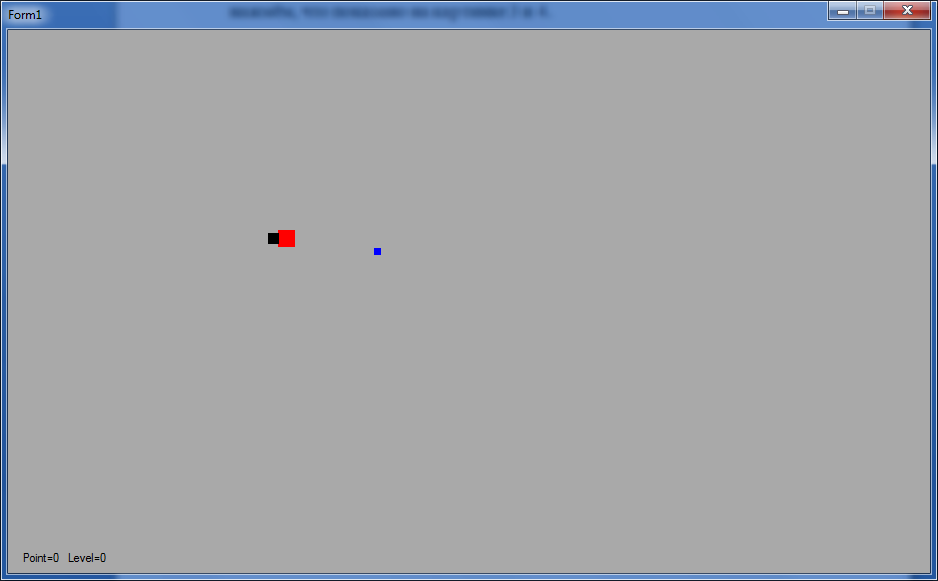


Картинка 2 – Начало игры

При нажатии на клавиши “*Arrow up*”, “*Arrow down*”, ”*Arrow left*”, ”*Arrow right*”. Наш объект “Змейка” будет двигаться в зависимости, куда мы нажмём, что показано на картинке 3 и 4.

****

Картинка 3 – Передвижение змейки



Картинка 4 – Передвижение змейки

В ходе выполнения лабораторной работы были подключены “*xml*” и “*xsd*” файлы. Для передачи цвета и позиции “Змейки” использовался “*xml*” файл, a для проверки правильности ввода данных в “*xml*” использовали “*xsd*” файл.

**Вывод:** Были получены навыки разработки программ-игр, использующих формы, элементы управления и принципы событийно-управляемого программирования.

**Приложение А**

**(C#)**

**Form1.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

//using System.Threading.Tasks;

using System.Windows.Forms;

namespace Laba\_5

{

public partial class Form1 : Form

{

Keys choose;

Timer timer;

Random rand = new Random();

int x1 = 0,

y1 = 0;

int x = 10

, y = 10;

public int X1

{

get { return x1; }

set { x1 = value; }

}

public int Y1

{

get { return y1; }

set { y1 = value; }

}

public Form1()

{

InitializeComponent();

}

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

{

this.FormBorderStyle = FormBorderStyle.FixedDialog;

this.StartPosition = FormStartPosition.CenterScreen;

}

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

{

timer = new Timer();

Graphics g = CreateGraphics();

button1.Visible = false;

//button1.Location = this.PointToClient(MousePosition);

g.Clear(Color.DarkGray);

X = 10;

Y = 10;

X1 = 0;

Y1 = 0;

CountPoint = 0;

level = 0;

choose = Keys.K;

ChangeMove = 0;

timer.Interval = 100;

timer.Start();

timer.Tick += new EventHandler(timer\_Tick);

}

public int X

{

get { return x; }

set { x = value; }

}

public int Y

{

get { return y; }

set { y = value; }

}

int changemove = 5;

public int ChangeMove

{

get { return changemove; }

set { changemove = value; }

}

int CountPoint = 0;

int level = 0;

Rectangle rect;

bool flag = true;

public void timer\_Tick(object sender, EventArgs e)

{

Graphics g1 = CreateGraphics();

g1.Clear(Color.DarkGray);

//------------------------------------------------

if (choose == Keys.Right)

{

if (ChangeMove == 3)

{

timer.Stop();

MessageBox.Show("Boom noob!,try again");

button1.Visible = true;

}

X += 10;

changemove = 0;

}

if (choose == Keys.Down)

{

if (ChangeMove == 2)

{

timer.Stop();

MessageBox.Show("Boom noob!,try again");

button1.Visible = true;

}

Y += 10;

changemove = 1;

}

if (choose == Keys.Up)

{

if (ChangeMove == 1)

{

timer.Stop();

MessageBox.Show("Boom noob!,try again");

button1.Visible = true;

}

Y -= 10;

changemove = 2;

}

if (choose == Keys.Left)

{

if (ChangeMove == 0)

{

timer.Stop();

MessageBox.Show("Boom noob!,try again");

button1.Visible = true;

}

X -= 10;

changemove = 3;

}

//int yc = MousePosition.X;

//int xc = MousePosition.Y;

//------------------------------------------------

if (X > 938 || Y > 581 || Y < 0 || X < 0)

{

timer.Stop();

MessageBox.Show("Boom noob!,try again");

button1.Visible = true;

}

//-------------------------------------------------

if (( X >= X1 - 12 && Y >= Y1 - 12) && (X <= X1 + 12 && Y <= Y1 + 12) )

{

flag = true;

if (CountPoint == 2)

{

CountPoint = 0;

level++;

if (timer.Interval == 20)

{

timer.Stop();

MessageBox.Show("Congratulations son,you win");

Application.Exit();

}

else

{

timer.Interval -= 20;

}

}

Point.Text = "Point=" + CountPoint + " " + "Level=" + level;

CountPoint++;

}

if (flag)

{

To4ka();

flag = false;

}

//----------------------------------------------------------

g1.DrawRectangle(Pens.Blue, rect = new Rectangle(X1, Y1, 6, 6));

g1.FillRectangle(Brushes.Blue, rect);

g1.DrawRectangle(Pens.Red, rect = new Rectangle(X, Y, 16, 16));

g1.FillRectangle(Brushes.Red, rect);

//-------------------------------------------------------------------------

if (ChangeMove == 1)

{

g1.DrawRectangle(Pens.Black, rect = new Rectangle(X + 3, Y - 10, 10, 10));

g1.FillRectangle(Brushes.Black, rect);

}

if (ChangeMove == 0)

{

g1.DrawRectangle(Pens.Black, rect = new Rectangle(X - 10, Y + 3, 10, 10));

g1.FillRectangle(Brushes.Black, rect);

}

if (ChangeMove == 3)

{

g1.DrawRectangle(Pens.Black, rect = new Rectangle(X + 16, Y + 3, 10, 10));

g1.FillRectangle(Brushes.Black, rect);

}

if (ChangeMove == 2)

{

g1.DrawRectangle(Pens.Black, rect = new Rectangle(X + 3, Y + 16, 10, 10));

g1.FillRectangle(Brushes.Black, rect);

}

//----------------------------------------------------------------------

}

public void To4ka()

{

X1 = rand.Next(0, 938 - 50);

Y1 = rand.Next(0, 581 - 50);

}

private void Form1\_KeyDown(object sender, KeyEventArgs e)

{

choose = e.KeyCode;

}

}

}

**XML файл:**

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

**<options>**

**<color>Orange</color>**

**<tick>150</tick>**

**<Position>**

**<x>50</x>**

**<y>50</y>**

**</Position>**

**</options>**

**XSD файл:**

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

**<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">**

**<xs:element name="options">**

**<xs:complexType>**

**<xs:sequence>**

**<xs:element name="color" type="string" />**

**<xs:element name="tick" type="tick\_int"/>**

**<xs:element name="Position">**

**<xs:complexType>**

**<xs:sequence>**

**<xs:element name="x" type="int"/>**

**<xs:element name="y" type="int"/>**

**</xs:sequence>**

**</xs:complexType>**

**</xs:element>**

**</xs:sequence>**

**</xs:complexType>**

**</xs:element>**

**<xs:simpleType name="string">**

**<xs:restriction base="xs:Name">**

**<xs:enumeration value="Orange"/>**

**<xs:enumeration value="Red"/>**

**<xs:enumeration value="Blue"/>**

**<xs:enumeration value="Black"/>**

**</xs:restriction>**

**</xs:simpleType>**

**<xs:simpleType name="tick\_int">**

**<xs:restriction base="xs:int">**

**<xs:minInclusive value="10"/>**

**<xs:maxInclusive value="200"/>**

**</xs:restriction>**

**</xs:simpleType>**

**<xs:simpleType name="int">**

**<xs:restriction base="xs:int">**

**<xs:minInclusive value="1"/>**

**<xs:maxInclusive value="100"/>**

**</xs:restriction>**

**</xs:simpleType>**

**</xs:schema>**