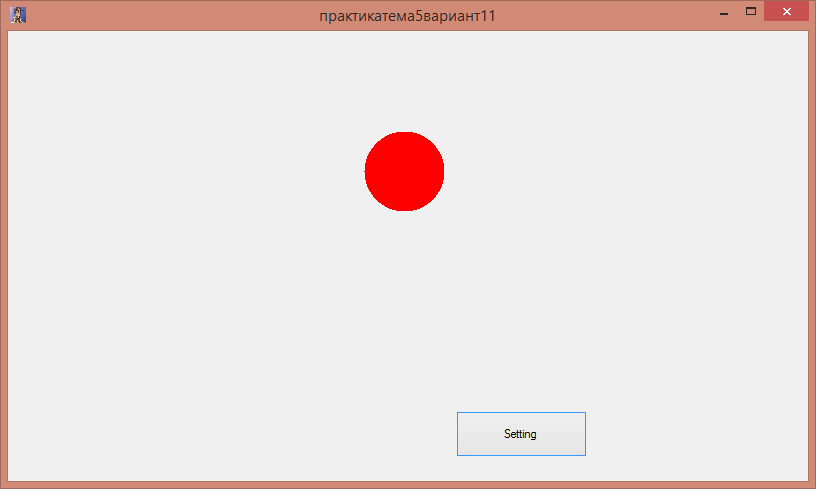
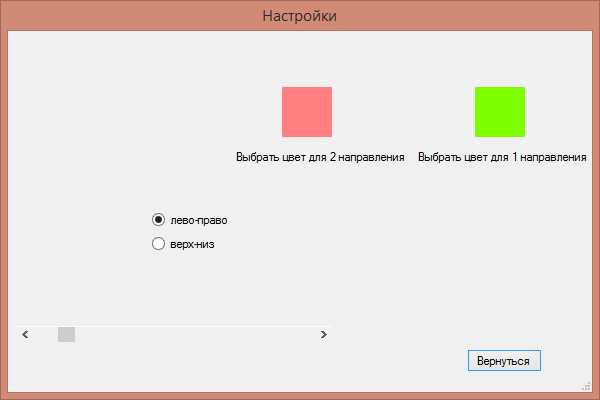
Отчет по 5 теме практики студента Лавриненко Алексей

# 5. Работа с реестром ОС Windows

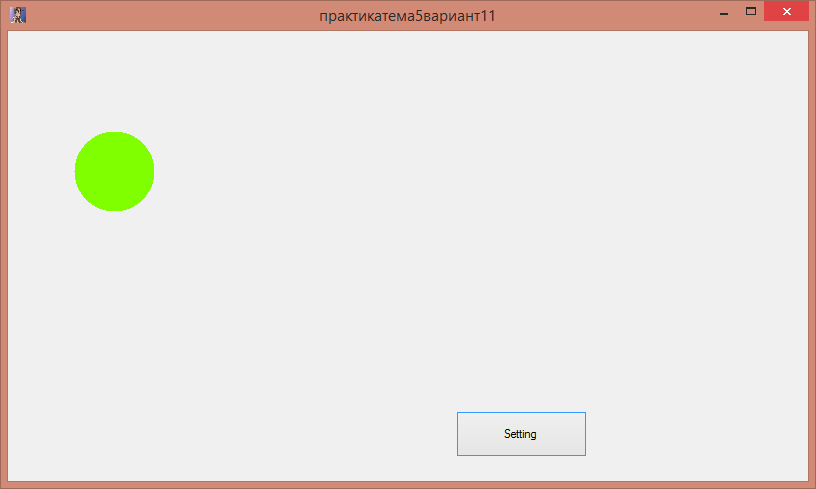
Открываем exeшник для 5 практической:



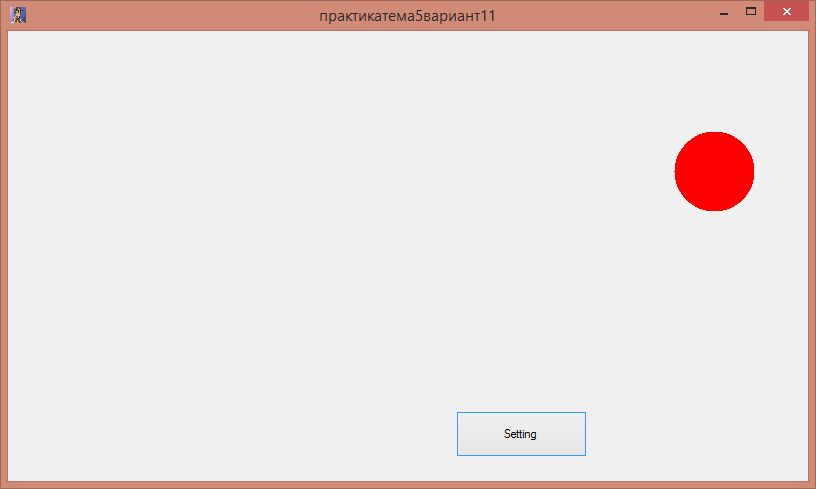
Устанавливаем самые пестрые цвета:



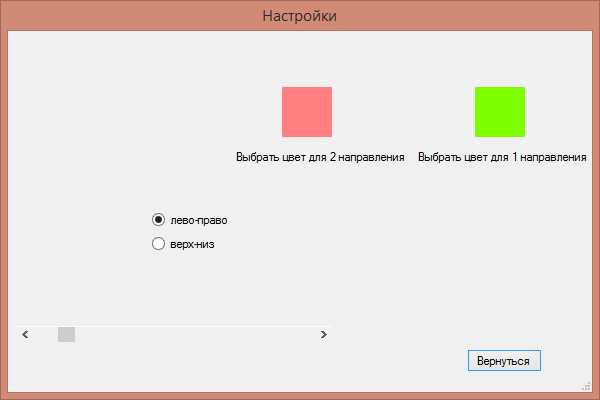
Закрываем:



Открываем вновь и видим, что шарик не перекрасился☹:

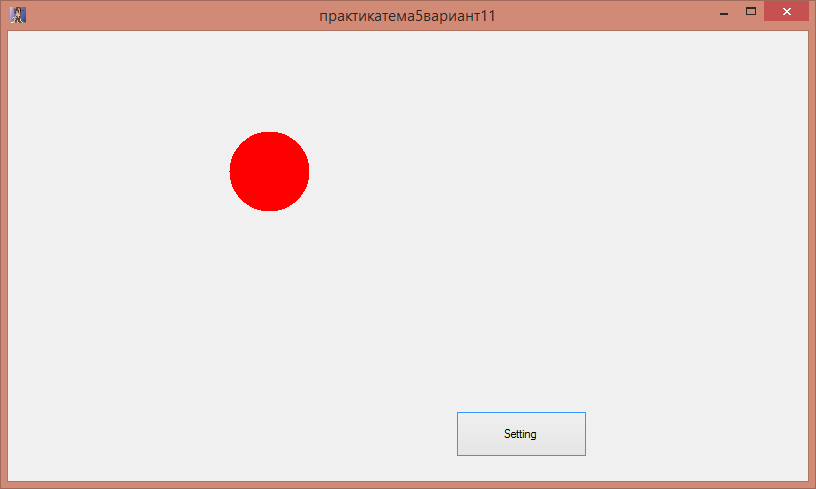


Однако в настройках всё указано как и должно быть:

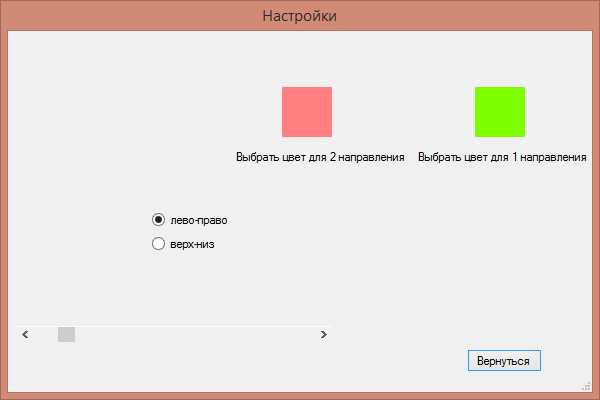


**Тема 5:** Работа с графикой

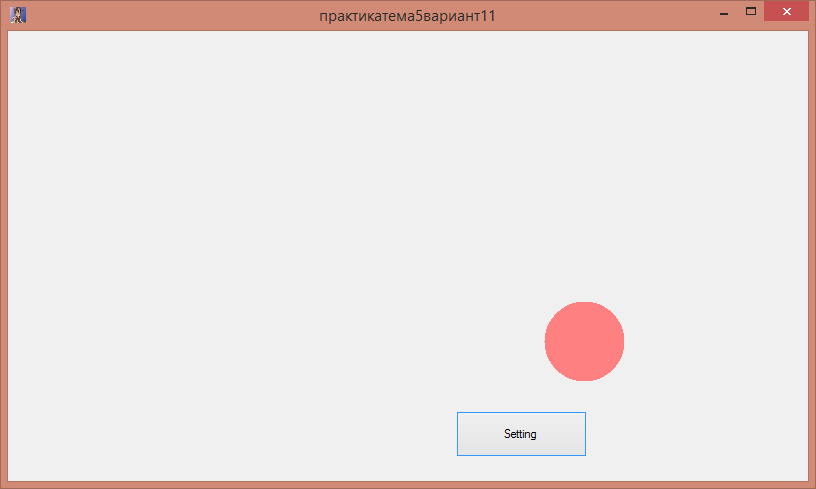
Внешний Exeшника:



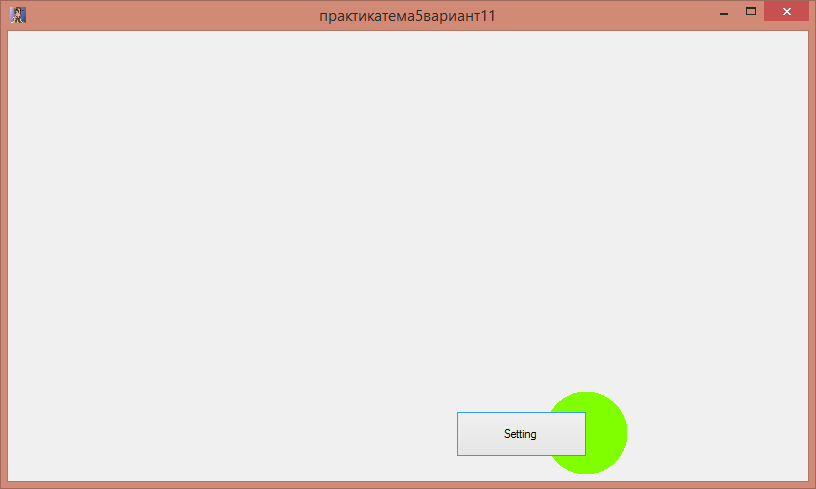
2 форма:



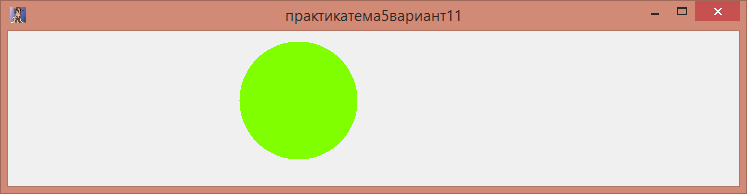
Переключим на режим «верх-низ»:

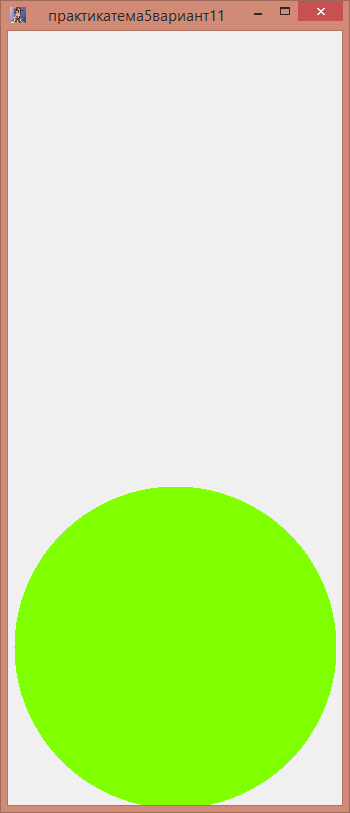


При достижении края, кружок меняет цвет и двигается в противоположном направлении:



Причём край платформы высчитывается в реальном времени:





Скрипт 1 формы:

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 WindowsFormsApp5

{

public partial class Form1 : Form

{

public Color color1 = Color.Red;

public Color color2 = Color.Red;

public bool right = true;

public bool down = false;

public int clock2 = 100;

public Form1()

{

InitializeComponent();

this.KeyPreview = true;

}

int sec = 0;

int w = 80, h = 80;

int x = 1, y = 100;

int dx = 5;

enum STATUS { Left, Right, Up, Down }; //направления движения

STATUS flag; //флаг изменения направления движения

//

SolidBrush brush = new SolidBrush(Color.FromArgb(Properties.Settings.Default.r, Properties.Settings.Default.g, Properties.Settings.Default.b)); // кисть

Rectangle rc; //прямоугольная область, в которой находиться фигура

private void timer1\_Tick(object sender, EventArgs e)

{

timer1.Interval = clock2;

sec++; // секунды

rc = new Rectangle(x, y, w, h); // размер прямоугольной области

this.Invalidate(rc, true); // вызываем прорисовку области

if (flag == STATUS.Left) // движение влево

{

x -= dx;

}

if (flag == STATUS.Right)// движение вправо

{

x += dx;

}

//можно попробовать с помощью буллов, завтра этим займусь

if (flag == STATUS.Up)

{

y -= dx;

}

if (flag == STATUS.Down)

{

y += dx;

}

//

if (x >= (this.ClientSize.Width - w)) // если достигли правого края формы

{

flag = STATUS.Left; // меняем статус движения на левый

// SolidBrush brush = new SolidBrush(color1);

}

else

if ((x <= 1)) // если достигли левого края формы

{

flag = STATUS.Right; // меняем статус движения на правый

// SolidBrush brush = new SolidBrush(color2);

}

//

if (y >= (this.ClientSize.Height - w)) // если достигли края формы

{

flag = STATUS.Up; // меняем статус движения

h++;

w++;

}

else if (y <= 1) // если достигли края формы

{

flag = STATUS.Down; // меняем статус движения

h++;

w++;

}

//

rc = new Rectangle(x, y, w, h); // новая прямоугольная область

this.Invalidate(rc, true); // вызываем прорисовку этой области

}

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

{

if (right)

{

if (flag == STATUS.Right)

flag = STATUS.Left;

else

flag = STATUS.Right;

}

if (down)

{

if (flag == STATUS.Up)

flag = STATUS.Down;

else

flag = STATUS.Up;

}

}

private void Form1\_Paint(object sender, PaintEventArgs e)

{

//

if (flag == STATUS.Right || flag == STATUS.Down)

{

brush.Color = color2;

}

if (flag == STATUS.Left || flag == STATUS.Up)

{

brush.Color = color1;

}

e.Graphics.FillEllipse(brush, rc); // рисуем закрашенный эллипс

}

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

{

Form setting = new Form2();

setting.Owner = this;

setting.Top = this.Top;

setting.ShowDialog();

//setting.color = color1;

//setting.color2 = color2;

}

private void hScrollBar1\_Scroll(object sender, ScrollEventArgs e)

{

//timer1.Interval = clock2;// hScrollBar1.Value;

}

public int MySpeed

{

get

{

return 100 - timer1.Interval;

}

set

{

timer1.Interval = 100 - value;

}

}

private void Form1\_KeyPress(object sender, KeyPressEventArgs e)

{

}

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

{

Form1 form = new Form1();

if (e.KeyCode == Keys.Escape)

{

//form.Show();

form.Close();

}

}

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

{

}

private void Form1\_ControlRemoved(object sender, ControlEventArgs e)

{

if (right)

{

if (flag == STATUS.Right)

flag = STATUS.Left;

else

flag = STATUS.Right;

}

if (down)

{

if (flag == STATUS.Up)

flag = STATUS.Down;

else

flag = STATUS.Up;

}

}

public Color MyColor

{

get

{

return brush.Color;

}

set

{

brush.Color = value;

}

}

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

{

if (e.KeyCode == Keys.Escape)

{

Application.Exit();

}

/\* Form1 form = new Form1();

if (e.KeyCode == Keys.Escape)

{

//form.Show();

form.Close();

}\*/

}

public void xpenb(bool right, bool down)

{

if (right)

{

if (flag == STATUS.Right)

flag = STATUS.Left;

else

flag = STATUS.Right;

}

if (down)

{

if (flag == STATUS.Up)

flag = STATUS.Down;

else

flag = STATUS.Up;

}

}

}

}

Скрипт 2 формы:

using System;

using Microsoft.Win32;

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 WindowsFormsApp5

{

public partial class Form2 : Form

{

public ColorDialog clr = new ColorDialog();

//Color color = Color.Red;

public Color color;

public ColorDialog clr2 = new ColorDialog();

// public Color lor;

//Color color2 = Color.Red;

public Color color2;

public int clock=1000;

public bool left;

public bool up;

public Form2()

{

InitializeComponent();

}

private void Form2\_Load(object sender, EventArgs e)

{

Form1 Glavnuk = (Form1)this.Owner;

color = Color.FromArgb(Properties.Settings.Default.r, Properties.Settings.Default.g, Properties.Settings.Default.b);

// color=Glavnuk.color1;

color2 = Color.FromArgb(Properties.Settings.Default.r2, Properties.Settings.Default.g2, Properties.Settings.Default.b2);

hScrollBar1.Value = Properties.Settings.Default.clock;

label3.BackColor = color2;

label1.BackColor = color;

}

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

{

SaveSet(color,color2,clock);

// Form Glavnuk = Application.OpenForms[0];

Form1 Glavnuk = (Form1)this.Owner;

Glavnuk.Top = this.Top;

// Glavnuk.StartPosition = FormStartPosition.Manual;

Glavnuk.Show();

Glavnuk.clock2= clock;

Glavnuk.color1 = color;

Glavnuk.color2 = color2;

left = radioButton1.Checked;

up = radioButton2.Checked;

//Glavnuk.right = left;

//Glavnuk.down = up;

Glavnuk.xpenb(left,up);

this.Hide();

}

private void hScrollBar1\_Scroll(object sender, ScrollEventArgs e)

{

//Form1.

//Timer1.

clock = hScrollBar1.Value;

}

private void label2\_Click(object sender, EventArgs e)

{

}

private void label1\_Click(object sender, EventArgs e)

{

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

{

color = clr.Color;

label1.BackColor = clr.Color;

}

}

private void label3\_Click(object sender, EventArgs e)

{

if (clr2.ShowDialog() == DialogResult.OK)

{

color2 = clr2.Color;

label3.BackColor = clr2.Color;

}

}

private void radioButton1\_CheckedChanged(object sender, EventArgs e)

{

}

private void radioButton2\_CheckedChanged(object sender, EventArgs e)

{

}

public void SaveSet(Color clr,Color clr2,int clock)

{

//Save color

int r = clr.R;

int g = clr.G;

int b = clr.B;

int r2 = clr2.R;

int g2 = clr2.G;

int b2 = clr2.B;

//save this in C:\\Users...

Properties.Settings.Default.r = r;

Properties.Settings.Default.g = g;

Properties.Settings.Default.b = b;

Properties.Settings.Default.r2 = r2;

Properties.Settings.Default.g2= g2;

Properties.Settings.Default.b2 = b2;

Properties.Settings.Default.clock =clock;

Properties.Settings.Default.Save();

/\* RegistryKey currentUserKey = Registry.CurrentUser;

RegistryKey Key = currentUserKey.CreateSubKey("Key");

//Save color

int r = clr.R;

int g = clr.G;

int b = clr.B;

int r2 = clr2.R;

int g2 = clr2.G;

int b2 = clr2.B;

//Key.SetValue("tr", tr);

//Save in pc

Key.SetValue("r", r);

Key.SetValue("g", g);

Key.SetValue("b", b);

Key.SetValue("r2", r2);

Key.SetValue("g2", g2);

Key.SetValue("b2", b2);

Key.Close();\*/

}

}

}