Федеральное агентство связи

Федеральное государственное бюджетное образовательное учреждение

высшее образования

«Сибирский Государственный Университет телекоммуникаций и информатики»

Лабораторная работа №6

Выполнил студент 2 курса

ИВТ, гр ИП-814

Якунин Андрей

Проверил

Ассистент ПМиК

Суходоева Наталья Николаевна

Новосибирск 2020

**Задание**

Перетаскивание графических объектов с помощью мыши (При нажатии левой кнопки мыши внутри границ фигуры и перемещении мыши, графический объект перетаскивается на новое место в соответствии с движением мыши).

Реализация перетаскивания любого объекта Shape возложена на три обработчика событий этого объекта: MouseDown, MouseMove и MouseUp.

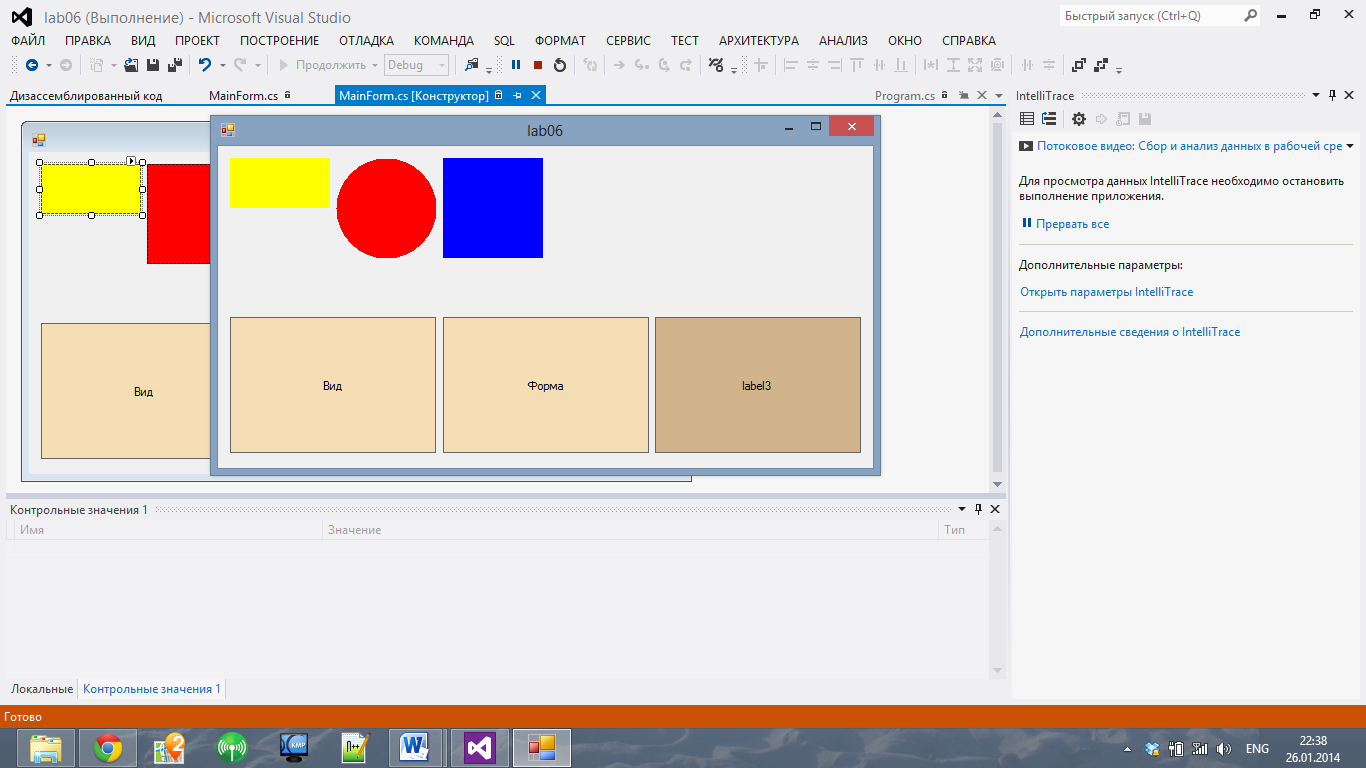
1. С помощью компонента PictureBox на палитре компонентов создайте на форме три фигуры: жёлтый прямоугольник со скругленными краями, красный круг и синий квадрат. Для выбора формы фигуры используйте свойство Shape, для определения цвета и шаблона заливки внутри области - свойство Brush, для определения цвета и стиля контура – свойство Pen. Для всех созданных объектов класса Shape используйте одни и те же обработчики событий.
2. В разделе private опишите переменную Dragging типа boolean, которая принимает значение True во время выполнения операции “щелкнуть и перетащить”. В обработчике инициализации формы установите значение этой переменной равным False.
3. В процедуре обработки события MouseDown объекта Shape следует выполнить следующие действия:
   * Установить флажок Dragging в True, если нажата левая кнопка мыши.
   * Нарисовать точечный прямоугольник для перетаскивания объекта Shape.
   * Сохранить координаты мыши в момент нажатия левой кнопки мыши для их последующего использования при перерисовке объекта Shape в новом месте в обработчике события MouseUp. Переменные опишите в разделе private.
   * Сохранить координаты мыши в момент нажатия левой кнопки мыши для определения начального положения мыши при перемещении для обработчика события MouseMove (эти переменные изменяют свои значения в этом обработчике событий). Переменные опишите в разделе private.
4. В процедуре обработки события MouseMove в случае, если установлен режим перетаскивания, необходимо выполнить следующие действия:
   * Установить в качестве начальных координат для следующего перемещения мыши текущие координаты мыши.
   * Определить заливку прямоугольника для перетаскивания.
5. Процедура обработки события MouseUp в случае, если установлен режим перетаскивания, должна выполнить следующее:
   * Сбросить флаг Dragging для отмены операции перетаскивания.
   * Удалить последний контур выделения объекта.
   * Переместить фигуру к её новому местоположению, изменив значения свойств Left и Top объекта Shape.

*Задание 2*. Реализация интерфейса Drag&Drop. Изменение формы фигуры при переносе

1. Поместите на форму 3 компонента Label, 2 компонента Shape (один из них квадрат, другой эллипс )

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вид |  | Форма |  | Красный квадрат |

1. Запрограммируйте работу обработчика события MouseEventArgs метки Форма так, чтобы принимались только компоненты типа Shape при перетаскивании компонента на метку
2. В обработчике события MouseEventArgs метки запрограммируйте изменение формы.(квадрат становится окружностью, окружность – квадратом )
3. Запрограммируйте работу обработчика события MouseEventArgs метки Вид так, чтобы принимались только компоненты типа Shape и в третьей метке появляется информация о цвете и форме фигуры при перетаскивании.



**Листинг**

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 WindowsFormsApplication2

{

public partial class Lab6 : Form

{

Rectangle Rectangle = new Rectangle(10, 10, 200, 100);

Rectangle Circle = new Rectangle(220, 10, 150, 150);

Rectangle Square = new Rectangle(380, 10, 150, 150);

bool x1 = false;

bool x2 = false;

bool x3 = false;

int i,i1,i2 = 0;

int j,j1,j2 = 0;

int X, Y, dX = -300, dY = - 300;

int LastClicked = 0;

public Lab6()

{

InitializeComponent();

}

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

{

}

private void pictureBox1\_Click(object sender, EventArgs e)

{

}

private void Lab6\_Load(object sender, EventArgs e)

{

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

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

{

}

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

{

}

private void pictureBox1\_Paint(object sender, PaintEventArgs e)

{

e.Graphics.FillEllipse(Brushes.Red, Circle);

e.Graphics.FillRectangle(Brushes.Blue, Square);

e.Graphics.FillRectangle(Brushes.Yellow, Rectangle);

if(x1)

{

e.Graphics.FillEllipse(Brushes.Red, Circle);

e.Graphics.FillRectangle(Brushes.Blue, Square);

e.Graphics.FillRectangle(Brushes.Yellow, Rectangle);

}

if(x2)

{

e.Graphics.FillRectangle(Brushes.Blue, Square);

e.Graphics.FillRectangle(Brushes.Yellow, Rectangle);

e.Graphics.FillEllipse(Brushes.Red, Circle);

}

if(x3)

{

e.Graphics.FillRectangle(Brushes.Yellow, Rectangle);

e.Graphics.FillEllipse(Brushes.Red, Circle);

e.Graphics.FillRectangle(Brushes.Blue, Square);

}

pictureBox1.Invalidate();

}

private void pictureBox1\_MouseDown(object sender, MouseEventArgs e)

{

if ((e.X < Rectangle.X + Rectangle.Width) && (e.X > Rectangle.X))

{

if ((e.Y < Rectangle.Y + Rectangle.Height) && (e.Y > Rectangle.Y))

{

x1 = true;

i = e.X - Rectangle.X;

j = e.Y - Rectangle.Y;

}

}

if ((e.X < Circle.X + Circle.Width) && (e.X > Circle.X))

{

if ((e.Y < Circle.Y + Circle.Height) && (e.Y > Circle.Y))

{

x2 = true;

i1 = e.X - Circle.X;

j1 = e.Y - Circle.Y;

}

}

if ((e.X < Square.X + Square.Width) && (e.X > Square.X))

{

if ((e.Y < Square.Y + Square.Height) && (e.Y > Square.Y))

{

x3 = true;

i2 = e.X - Square.X;

j2 = e.Y - Square.Y;

}

}

}

private void pictureBox1\_MouseUp(object sender, MouseEventArgs e)

{

if (LastClicked == 2)

{

if ((label2.Location.X < Circle.X + Circle.Width) && (label2.Location.X > Circle.X))

{

if ((label2.Location.Y < Circle.Y + Circle.Height) && (label2.Location.Y > Circle.Y))

{

X = Circle.X;

Y = Circle.Y;

dX = i1;

dY = j1;

Circle.X = Square.X;

Circle.Y = Square.Y;

i1 = i2;

j1 = j2;

Square.X = X;

Square.Y = Y;

i2 = dX;

j2 = dY;

}

}

}

if (LastClicked == 3)

{

if ((label2.Location.X < Square.X + Square.Width) && (label2.Location.X > Square.X))

{

if ((label2.Location.Y < Square.Y + Square.Height) && (label2.Location.Y > Square.Y))

{

X = Square.X;

Y = Square.Y;

dX = i2;

dY = j2;

Square.X = Circle.X;

Square.Y = Circle.Y;

i2 = i1;

j2 = j1;

Circle.X = X;

Circle.Y = Y;

i1 = dX;

j1 = dY;

}

}

}

x1 = false;

x2 = false;

x3 = false;

}

private void pictureBox1\_MouseMove(object sender, MouseEventArgs e)

{

if (x1)

{

Rectangle.X = e.X - i;

Rectangle.Y = e.Y - j;

pictureBox1.Invalidate();

LastClicked = 1;

}

if (x2)

{

Circle.X = e.X - i1;

Circle.Y = e.Y - j1;

pictureBox1.Invalidate();

LastClicked = 2;

}

if (x3)

{

Square.X = e.X - i2;

Square.Y = e.Y - j2;

pictureBox1.Invalidate();

LastClicked = 3;

}

if ((label1.Location.X < Square.X + Square.Width) && (label1.Location.X > Square.X))

{

if ((label1.Location.Y < Square.Y + Square.Height) && (label1.Location.Y > Square.Y))

{

label3.Text = "Синий квадрат";

}

}

if ((label1.Location.X < Circle.X + Circle.Width) && (label1.Location.X > Circle.X))

{

if ((label1.Location.Y < Circle.Y + Circle.Height) && (label1.Location.Y > Circle.Y))

{

label3.Text = "Красный круг";

}

}

if ((label1.Location.X < Rectangle.X + Rectangle.Width) && (label1.Location.X > Rectangle.X))

{

if ((label1.Location.Y < Rectangle.Y + Rectangle.Height) && (label1.Location.Y > Rectangle.Y))

{

label3.Text = "Желтый прямоугольник";

}

}

pictureBox1.Invalidate();

}

private void label2\_MouseUp(object sender, MouseEventArgs e)

{

}

}

}

**Скриншоты**

