«**Разработка интерфейса настройки системы управления с помощью табличного редактора.**»

Выполнил:

Вырмаскин Н.А

Набережные Челны

2015

***Задача:***

Разработать графический интерфейс для отображения входных воздействий на исполнительный механизм и программу, импортирующую данные с табличного редактора и отправляющие их на порт клавиатуры вызывая загорания и потухание светодиодов отражающих нажатие клавиш «Num Lock», «Caps Lock», «Scroll Lock», в зависимости от заданных входных сигналов.

***Вид программы:***

Сперва вид программы имеет вид:



После нажатия кнопки отобразить:



Надпись на кнопке меняется на очистит, появляется разметка и отображение областей соответствующих кнопке. Так же появляется отображение работы кнопок. Одна квадратная ячейка занимает 100 мсек. Т.е. для того чтобы интервал между изменениями нажатия кнопок был 1сек. Точки нужно ставить в 10 клетках друг от друга.

***Способ работы программы:***

1. Выбирается тип линии построения графика: линии или сплайн
2. Нажатиями мышкой по pictureBox строится по точкам график.
3. Точки заносятся в массив данных.
4. После того как последняя точка достигла конца pictureBox, массив выводится в dataGridView и кнопка начать становится доступной.
5. После нажатия на кнопку начать, мигают светодиоды на соответствующих кнопках и это так же отображается изменением цвета с красного на зелёный в картинках под названиями кнопок.
6. После выполнения эти действия можно повторить снова нажав кнопку начать или создать другой массив данных нажав кнопку очистить и повторив действия с 2-5 п.



***Код программы:***

***Основной класс «One»:***

class One

 {

 public static int MouseX { get; set; } //Свойство получающее координаты нажатия мыши по оси x

 public static int MouseY { get; set; } //Свойство получающее координаты нажатия мыши по оси y

 public static int X { get; set; } //Свойство получающее условную координату нажатия мыши по оси x

 public static int Y { get; set; } //Свойство получающее условную координату нажатия мыши по оси y

 public static int[] Points\_x { get; set; } //Содержит массив условных координат x для занисения в dataGridView

 public static int[] Points\_y { get; set; } //Содержит массив условных координат y для занисения в dataGridView

 public void Activate\_arr() //Очищает данные

 {

 X = 0;

 Y = 0;

 Points\_x = null;

 Points\_y = null;

 if (Points\_x == null && Points\_y == null)

 {

 int[] r = new int[1];

 Points\_x = r;

 Points\_y = r;

 Points\_x[0] = 120;

 Points\_y[0] = 0;

 }

 }

 public void Add\_arr(int x, int y) //Добавляяет данные в массив

 {

 if(x > X)

 {

 X = x;

 Y = y;

 int[] arr\_x = new int[Points\_x.Length + 1];

 int[] arr\_y = new int[Points\_x.Length + 1];

 for (int i = 0; i < Points\_x.Length; i++)

 {

 arr\_x[i] = Points\_x[i];

 arr\_y[i] = Points\_y[i];

 }

 arr\_x[Points\_x.Length] = x;

 arr\_y[Points\_y.Length] = y;

 Points\_x = arr\_x;

 Points\_y = arr\_y;

 }

 }

 public int Cheked(int a) //Переводит полученную координату в условную

 {

 int b = 0;

 if(((Math.Abs(a/10) \*10) +5) < a)

 {

 b = (Math.Abs(a / 10) \* 10) + 10;

 }

 else

 {

 b = (Math.Abs(a / 10) \* 10);

 }

 return b;

 }

 [DllImport("user32.dll")] //Показывает, что в качестве статической точки входа неуправляемая динамическая библеотека (DLL) предоставляет метод с атрибутами.

 static extern void keybd\_event(byte bVk, byte bScan, uint dwFlags, UIntPtr dwExtraInfo); //Тот самый метод с атрибутами

 public void Caps\_Lock() //метод изменения нажатия кнопки.

 {

 const int KEYEVENTF\_EXTENDEDKEY = 0x1;

 const int KEYEVENTF\_KEYUP = 0x2;

 keybd\_event(0x14, 0x45, KEYEVENTF\_EXTENDEDKEY, (UIntPtr)0);

 keybd\_event(0x14, 0x45, KEYEVENTF\_EXTENDEDKEY | KEYEVENTF\_KEYUP, (UIntPtr)0);

 }

 public void Num\_Lock() //метод изменения нажатия кнопки.

 {

 const int KEYEVENTF\_EXTENDEDKEY = 0x1;

 const int KEYEVENTF\_KEYUP = 0x2;

 keybd\_event(0x90, 0x45, KEYEVENTF\_EXTENDEDKEY, (UIntPtr)0);

 keybd\_event(0x90, 0x45, KEYEVENTF\_EXTENDEDKEY | KEYEVENTF\_KEYUP, (UIntPtr)0);

 }

 public void Scroll\_Lock() //метод изменения нажатия кнопки.

 {

 const int KEYEVENTF\_EXTENDEDKEY = 0x1;

 const int KEYEVENTF\_KEYUP = 0x2;

 keybd\_event(0x91, 0x45, KEYEVENTF\_EXTENDEDKEY, (UIntPtr)0);

 keybd\_event(0x91, 0x45, KEYEVENTF\_EXTENDEDKEY | KEYEVENTF\_KEYUP, (UIntPtr)0);

 }

 }

***Главная форма «Form1»:***

public partial class Form1 : Form

 {

 One o = new One();

 public Form1()

 {

 InitializeComponent();

 }

 async Task KeyboardDisco() //Ассинхронный метод изменения нажатий кнопок в зависимости от данных в массивах.

 {

 for (int i = 1; i < One.Points\_x.Length; i++)

 {

 if (One.Points\_y[i] <= 80)

 {

 int time = (One.Points\_x[i] - One.Points\_x[i - 1]) \* 10;

 pictureBox2.Image = imageList1.Images[1];

 pictureBox3.Image = imageList1.Images[0];

 pictureBox4.Image = imageList1.Images[0];

 o.Num\_Lock();

 await Task.Delay(time); //Создаёт задачу, которая будет выполнена после некоторой временной задержки.

 }

 if (One.Points\_y[i] <= 160 && One.Points\_y[i] > 80)

 {

 int time = (One.Points\_x[i] - One.Points\_x[i - 1]) \* 10;

 pictureBox3.Image = imageList1.Images[1];

 pictureBox2.Image = imageList1.Images[0];

 pictureBox4.Image = imageList1.Images[0];

 o.Caps\_Lock();

 await Task.Delay(time); //Создаёт задачу, которая будет выполнена после некоторой временной задержки.

 }

 if (One.Points\_y[i] <= 240 && One.Points\_y[i] > 160)

 {

 int time = (One.Points\_x[i] - One.Points\_x[i - 1]) \* 10;

 pictureBox4.Image = imageList1.Images[1];

 pictureBox2.Image = imageList1.Images[0];

 pictureBox3.Image = imageList1.Images[0];

 o.Scroll\_Lock();

 await Task.Delay(time); //Создаёт задачу, которая будет выполнена после некоторой временной задержки.

 }

 }

 }

 public void Draw\_line() //Чертит точку и типовую линию в условных координатах.

 {

 Pen pen = new Pen(Color.Red, 2);

 Graphics g = Graphics.FromHwnd(pictureBox1.Handle);

 int length = One.Points\_x.Length;

 if(radioButton2.Checked == true)

 {

 g.DrawLine(pen, One.Points\_x[length - 2], One.Points\_y[length - 2], One.Points\_x[length - 1], One.Points\_y[length - 1]);

 }

 if(radioButton1.Checked == true)

 {

 g.DrawBezier(pen, One.Points\_x[length - 2], One.Points\_y[length - 2], One.Points\_x[length - 2] + 20, One.Points\_y[length - 2], One.Points\_x[length - 1] - 20, One.Points\_y[length - 1], One.Points\_x[length - 1], One.Points\_y[length - 1]);

 }

 }

 public void Draw() //Метод начальной отрисовки pictureBox

 {

 Graphics g = Graphics.FromHwnd(pictureBox1.Handle);

 Brush br1 = new SolidBrush(Color.AntiqueWhite);

 Brush br2 = new SolidBrush(Color.FloralWhite);

 Brush br3 = new SolidBrush(Color.NavajoWhite);

 //Заполняет области кнопок разными чветами.

 g.FillRectangle(br1, 0, 0, 560, 80);

 g.FillRectangle(br2, 0, 80, 560, 160);

 g.FillRectangle(br3, 0, 160, 560, 240);

 Pen pen1 = new Pen(Color.Gray, 1);

 Pen pen2 = new Pen(Color.Black, 2);

 int i = 0;

 int j = 0;

 while(i <= 560 || j <= 240) //Чертит сетку

 {

 g.DrawLine(pen1, i ,240 ,i ,0);

 g.DrawLine(pen1, 0, j, 560, j);

 i += 10;

 j += 10;

 }

 //Чертит оси координат

 g.DrawLine(pen2, 1, 0, 1, 240);

 g.DrawLine(pen2, 0, 120, 560, 120);

 }

 private void pictureBox1\_MouseDown(object sender, MouseEventArgs e) //Метод обрабатывающий нажатие мышью в pictureBox

 {

 Pen pen = new Pen(Color.Red, 2);

 Graphics g = Graphics.FromHwnd(pictureBox1.Handle);

 //Заносит полученные координаты наатия

 One.MouseX = e.X;

 One.MouseY = e.Y;

 o.Add\_arr(o.Cheked(e.X), o.Cheked(e.Y));

 Draw\_line();

 g.DrawEllipse(pen, One.X - 1, One.Y - 1, 3, 3);

 if(One.MouseX > 555) //Выполняет отрисовку в dataGridView если график достиг конца pictureBox

 {

 dataGridView1.RowCount = One.Points\_x.Length;

 dataGridView1.ColumnCount = 3;

 for (int i = 0; i < One.Points\_x.Length; i++) //Выводит значения из массива в dataGridView1

 {

 dataGridView1.Rows[i].Cells[0].Value = i;

 dataGridView1.Rows[i].Cells[1].Value = One.Points\_x[i];

 dataGridView1.Rows[i].Cells[2].Value = One.Points\_y[i];

 }

 button2.Enabled = true;

 }

 }

 private void button1\_Click(object sender, EventArgs e) //Кнопка Отобразить(Очистить)

 {

 Graphics g = Graphics.FromHwnd(pictureBox1.Handle);

 g.Clear(Color.White);

 Draw();

 pictureBox2.Image = imageList1.Images[0];

 pictureBox3.Image = imageList1.Images[0];

 pictureBox4.Image = imageList1.Images[0];

 o.Activate\_arr();

 button1.Text = "Очистить";

 dataGridView1.Rows.Clear();

 button2.Enabled = false;

 }

 async void button2\_Click(object sender, EventArgs e) //Кнопка начать

 {

 var button = sender as Button; //Представляет элемент управления

 button.Enabled = false; //Делает кнопку не активной

 try { await KeyboardDisco(); } //Выполняет асинхронный метод

 finally { button.Enabled = true; } //По окончании делает кнопку активной

 }

 }

***Вывод:***

***Предоставили способ представления параметров настройки системы управления с помощью небольшого приложения типа построения графика и отображения в таблице.***

***Для удобства работы пользователя в технологических процессах применяются системы с графическими интерфейсами. В данном интерфейсе при построении графика пользователь преследует цель создать входное воздействие определённой амплитуды через определённый промежуток времени и отправить его на исполняемый метод. Исполняемым методом является асинхронный метод оперирующий нажатием кнопок.***