2021

Manual De Desarrollador

RODRIGO VASQUEZ

INGENIERIA EN SISTEMAS UMG | Antigua Guatemala

ÍNDICE

1 FormMenuPrincipal

1.1 FormMenuPrincipal

```
public FormMenuPrincipal()
{
    InitializeComponent();
    //Estas lineas eliminan los parpadeos del formulario o controles en la interfaz grafica (Pero no en un 100%)
    this.SetStyle(ControlStyles.ResizeRedraw, true);
    this.DoubleBuffered = true;
    label2.Visible = false;

}
```

1.2 WindProc:

```
protected override void WndProc(ref Message m)
{
    switch (m.Msg)
    {
        case WM_NCHITTEST:
            base.WndProc(ref m);
            var hitPoint = this.PointToClient(new Point(m.LParam.ToInt32() & 0xffff, m.LParam.ToInt32() >> 16));
            if (sizeGripRectangle.Contains(hitPoint))
                m.Result = new IntPtr(HTBOTTOMRIGHT);
            break;
        default:
            base.WndProc(ref m);
            break;
    }
}
```

1.3 OnSizeChanged:

```
protected override void OnSizeChanged(EventArgs e)
{
    base.OnSizeChanged(e);
    var region = new Region(new Rectangle(0, 0, this.ClientRectangle.Width, this.ClientRectangle.Height));

    sizeGripRectangle = new Rectangle(this.ClientRectangle.Width - tolerance, this.ClientRectangle.Height - tolerance, tolerance, tolerance);

    region.Exclude(sizeGripRectangle);
    this.panelContenedorPrincipal.Region = region;
    this.Invalidate();
}
```

1.4 OnPaint:

```
protected override void OnPaint(PaintEventArgs e)
{

    SolidBrush blueBrush = new SolidBrush(Color.FromArgb(55, 61, 69));
    e.Graphics.FillRectangle(blueBrush, sizeGripRectangle);

    base.OnPaint(e);
    ControlPaint.DrawSizeGrip(e.Graphics, Color.Transparent, sizeGripRectangle);
}
```

1.5 PANELBARRATITULO_MOUSEDONW:

```
private void PanelBarraTitulo_MouseDown(object sender, MouseEventArgs e)
{
    ReleaseCapture();
    SendMessage(this.Handle, 0x112, 0xf012, 0);
}
```

1.6 BTNMAXIMIZAR:

```
private void btnMaximizar_Click(object sender, EventArgs e)
{
    lx = this.Location.X;
    ly = this.Location.Y;
    sw = this.Size.Width;
    sh = this.Size.Height;
    this.Size = Screen.PrimaryScreen.WorkingArea.Size;
    this.Location = Screen.PrimaryScreen.WorkingArea.Location;
    btnMaximizar.Visible = false;
    btnNormal.Visible = true;

}
```

1.7 BTNNORMAL_CLICK:

```
private void btnNormal_Click(object sender, EventArgs e)
{
    this.Size = new Size(sw, sh);
    this.Location = new Point(lx, ly);
    btnNormal.Visible = false;
    btnMaximizar.Visible = true;
}
```

1.8 BTNMINIMIZAR_CLICK:

```
private void btnMinimizar_Click(object sender, EventArgs e)
{
    this.WindowState = FormWindowState.Minimized;
}

private void btnCerrar_Click(object sender, EventArgs e)
{
    if (Titulo.Text.Contains("*")) {
        if (MessageBox.Show("¿Está seguro de cerrar?", "Alerta", MessageBoxButtons.YesNo) == DialogResult.Yes)
        {
            if (MessageBox.Show("¿Desea Guardar antes de salir?", "Alerta", MessageBoxButtons.YesNo) == DialogResult.Yes)
            {
                save();
                Application.Exit();
            }
            else
            {
                Application.Exit();
            }
        }
    }
    else
    {

            Application.Exit();

    }
}
```

1.9 btnMenu_Click:

```
private void btnMenu_Click(object sender, EventArgs e)
{
    //-------CON EFECTO SLIDING
    if (panelMenu.Width == 230)
    {
        this.tmContraerMenu.Start();
    }
    else if (panelMenu.Width == 55)
    {
        this.tmExpandirMenu.Start();
    }

    //-------SIN EFECTO
    //if (panelMenu.Width == 55)
    //{
    //    panelMenu.Width = 230;
    //}
    //else

    //    panelMenu.Width = 55;
}
```

1.10 tmExpandirMenu_Tick:

```
private void tmExpandirMenu_Tick(object sender, EventArgs e)
{
    if (panelMenu.Width >= 230)
        this.tmExpandirMenu.Stop();
    else
        panelMenu.Width = panelMenu.Width + 5;

}
```

1.11 tmContraerMenu_Tick:

```
private void tmContraerMenu_Tick(object sender, EventArgs e)
{
    if (panelMenu.Width <= 55)
        this.tmContraerMenu.Stop();
    else
        panelMenu.Width = panelMenu.Width - 5;
}
```

1.12 aBrirToolStripMenuItem_Click:

```
private void aBRIRToolStripMenuItem_Click(object sender, EventArgs e)
{
    OpenFileDialog Openfile = new OpenFileDialog();
    Openfile.Filter = "Texto|*.HTML";
    if (Openfile.ShowDialog() == DialogResult.OK)
    {
        archivo = Openfile.FileName;
        using (StreamReader sr = new StreamReader(archivo))
        {
            fastColoredTextBox1.Text = sr.ReadToEnd();
        }
        NamesFiles(Openfile.FileName);

    }
}
```

1.13 NAMESFILES:

```
private void NamesFiles(string Nombre)
{
    char[] delimiterChars = { '.', '\\' };
    string[] nombre = Nombre.Split(delimiterChars);
    int cantidad = 0;
    cantidad = nombre.Length;
    Titulo.Text = nombre[cantidad - 2];
    label2.Visible = true;
}
```

1.14 CERRARARCHIVO_CLICK:

```
private void cerrarArchivo_Click(object sender, EventArgs e)
{
    fastColoredTextBox1.Clear();
    Titulo.Text = "Sin Titulo";
    label2.Visible = false;
    archivo = null;

}
```

1.15 NUEVOTOOLSSTRIPMENUITEM:

```
private void nUEVOToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (Titulo.Text.Contains("*"))
    {
        DialogResult dialogo = MessageBox.Show("¿Desea Guardar los Cambios?", "Guardar Cambios",
          MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (dialogo == DialogResult.No)
        {
            label2.Visible = false;
            fastColoredTextBox1.Clear();
            Titulo.Text = "Sin Titulo*";
            archivo = null;
        }
        else
        {
            save();
            label2.Visible = false;
            Titulo.Text = "Sin Titulo*";
            fastColoredTextBox1.Clear();
            archivo = null;
        }
    }
    else
    {
        label2.Visible = false;
        Titulo.Text = "Sin Titulo*";
        fastColoredTextBox1.Clear();
        archivo = null;
    }
}
```

1.16 SAVE:

```
private void save()//GUARDAR
{
    SaveFileDialog SaveFile = new SaveFileDialog();
    SaveFile.Filter = "Texto|*.HTML";
    if (archivo != null)
    {
        using (StreamWriter sw = new StreamWriter(archivo))
        {
            sw.Write(fastColoredTextBox1.Text);
        }
        Titulo.Text= Titulo.Text.Remove(Titulo.Text.Length - 1,1 );
    }
    else
    {
        if (SaveFile.ShowDialog() == DialogResult.OK)
        {
            archivo = SaveFile.FileName;
            using (StreamWriter sw = new StreamWriter(SaveFile.FileName))
            {
                sw.Write(fastColoredTextBox1.Text);
            }
        NamesFiles(archivo);
        }
    }


}
```

1.17 gUARDARToolStripMenuItem:

```
private void gUARDARToolStripMenuItem_Click(object sender, EventArgs e)
{
    save();
}

private void fastColoredTextBox1_TextChanged(object sender, FastColoredTextBoxNS.TextChangedEventArgs e)
{
    if (!Titulo.Text.Contains("*"))
    {
        Titulo.Text += "*";
    }

}
```

1.18 label2_Click:

```
private void label2_Click_1(object sender, EventArgs e)
{
    fastColoredTextBox1.Clear();
    Titulo.Text = "Sin Titulo*";
    label2.Visible = false;
    archivo = null;
    if (h != null)
    {


        h.Close();

        }
    if (dom!=null)
    {
        dom.Close();
    }


}
```

1.19 bUSCARToolStipMenuItem:

```
private void bUSCARToolStripMenuItem_Click(object sender, EventArgs e)
{
    fastColoredTextBox1.ShowFindDialog();
}

private void iMPRIMIRToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (printPreviewDialog1.ShowDialog() == DialogResult.OK)
    {
        printDocument1.Print();
    }
}
```

1.20 printDocument1:

```
private void printDocument1_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)
{
    e.Graphics.DrawString(fastColoredTextBox1.Text, new Font("Arial", 14, FontStyle.Bold), Brushes.Black, new PointF(100, 100));
}
```

1.21 FASTCOLOREDTEXTBOX:

```
private void fastColoredTextBox1_KeyDown(object sender, KeyEventArgs e)
{
    int cont_abre_etiqueta = 0;
    int cont_cierra_etiqueta = 0;
    string texto = fastColoredTextBox1.Text;
    char[] array = texto.ToCharArray();//separa en letras
    string error = "Cierre todas las etiquetas";
    string[] stringSeparators = new string[] { " ", "<", ">" };
    string[] array2 = texto.Split(stringSeparators, StringSplitOptions.None);//separa las palabras
    string busc = "";

    if (e.KeyCode == Keys.Up)
    {
        foreach (var p in array2)
        {
            busc = p;
            for (int i = 0; i < Reservadas.Length; i++)//para comparar cuando se abre la etiqueta
            {
                if (busc == Reservadas[i].ToString())
                {
                    cont_abre_etiqueta++;
                    int a = fastColoredTextBox1.Text.IndexOf(busc);


                }
            }
            for (int j = 0; j < Cierre_Reservadas.Length; j++)//para comparar cuando se cierra la etiqueta
            {
                if (busc == Cierre_Reservadas[j].ToString())
                {
                    cont_cierra_etiqueta++;
                }
            }
            busc = "";
        }

        if (cont_abre_etiqueta > 0 && cont_cierra_etiqueta > 0)
        {
            if (cont_abre_etiqueta == cont_cierra_etiqueta)
            {
                variableCom = fastColoredTextBox1.Text;
                if (dom != null)
                {
                    if (Application.OpenForms["DOM"] == null)
                    {

                        if (!dom.IsDisposed)
                        {
                            dom.Refresh();
                            dom.Activate();
                            dom.Show();
                            this.Focus();
                        }
```

```
                        this.Focus();
                    }
                    else
                    {
                        dom = new DOM();
                        dom.Refresh();
                        dom.Activate();
                        dom.Show();
                        this.Focus();
                    }
                }
                else
                {
                    dom.Refresh();
                    dom.Activate();
                    dom.Show();
                    this.Focus();
                }
            }
            else
            {
                dom = new DOM();
                dom.Refresh();
                dom.Activate();
                dom.Show();
                this.Focus();
            }
        }
        else
        {
            MessageBox.Show(error);
        }
    }
}

}
```

1.22 FASTCOLOREDTEXTBOX:

```
private void fastColoredTextBox1_KeyUp(object sender, KeyEventArgs e)
{
    int cont_abre_etiqueta = 0;
    int cont_cierra_etiqueta = 0;
    string texto = fastColoredTextBox1.Text;
    char[] array = texto.ToCharArray();//separa en letras
    string error = "Cierre todas las etiquetas";
    string[] stringSeparators = new string[] { " ", "<", ">" };
    string[] array2 = texto.Split(stringSeparators, StringSplitOptions.None);//separa las palabras
    string busc = "";

    if (e.KeyCode == Keys.Delete)
    {
        foreach (var p in array2)
        {
            busc = p;
            for (int i = 0; i < Reservadas.Length; i++)//para comparar cuando se abre la etiqueta
            {
                if (busc == Reservadas[i].ToString())
                {
                    cont_abre_etiqueta++;
                    int a = fastColoredTextBox1.Text.IndexOf(busc);


                }
            }
            for (int j = 0; j < Cierre_Reservadas.Length; j++)//para comparar cuando se cierra la etiqueta
            {
                if (busc == Cierre_Reservadas[j].ToString())
                {
                    cont_cierra_etiqueta++;
                }
            }
            busc = "";
        }

        if (cont_abre_etiqueta > 0 && cont_cierra_etiqueta > 0)
        {
            if (cont_abre_etiqueta == cont_cierra_etiqueta)
            {
                variableCom = fastColoredTextBox1.Text;
                if (dom != null)
                {
                    if (Application.OpenForms["DOM"] == null)
                    {

                        if (!dom.IsDisposed)
                        {
                            dom.Refresh();
                            dom.Activate();
                            dom.Show();
                            this.Focus();
                        }
                        else
```

```
                        else
                        {
                            dom = new DOM();
                            dom.Refresh();
                            dom.Activate();
                            dom.Show();
                            this.Focus();
                        }
                    }
                    else
                    {
                        dom.Refresh();
                        dom.Activate();
                        dom.Show();
                        this.Focus();
                    }
                }
                else
                {
                    dom = new DOM();
                    dom.Refresh();
                    dom.Activate();
                    dom.Show();
                    this.Focus();
                }
            }
            else
            {
                MessageBox.Show(error);
            }
        }
    }
}
int s1 = 0, s2 = 0, s3 = 0;

private void adelanteToolStripMenuItem_Click(object sender, EventArgs e)
{
    fastColoredTextBox1.Redo();
}

private void copiarToolStripMenuItem_Click(object sender, EventArgs e)
{
    fastColoredTextBox1.Copy();
}

private void cortarToolStripMenuItem_Click(object sender, EventArgs e)
{
    fastColoredTextBox1.Cut();
}

private void pEGARToolStripMenuItem1_Click(object sender, EventArgs e)
{
    fastColoredTextBox1.Paste();
```

1.23 pegarToolStripMenuItem1:

```
private void pEGARToolStripMenuItem1_Click(object sender, EventArgs e)
{
    fastColoredTextBox1.Paste();
}

private void sELECIONARTODOToolStripMenuItem_Click(object sender, EventArgs e)
{
    fastColoredTextBox1.SelectAll();
}

private void eLIMINARTODOToolStripMenuItem_Click(object sender, EventArgs e)
{
    fastColoredTextBox1.Clear();
}

private void rEEMPLAZARToolStripMenuItem_Click(object sender, EventArgs e)
{
    fastColoredTextBox1.ShowReplaceDialog();
}

private void fUENTEToolStripMenuItem_Click(object sender, EventArgs e)
{
    var formato = fontDialog1.ShowDialog();
    if (formato == DialogResult.OK)
    {
        fastColoredTextBox1.Font = fontDialog1.Font;
    }
}

private void iRAToolStripMenuItem1_Click(object sender, EventArgs e)
{
    fastColoredTextBox1.ShowGoToDialog();


}
```

1.24 pppToolStripMenuItem:

```
public Navegador h;
private void pppToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (h == null)
    {
        string content = "";
        content = fastColoredTextBox1.Text;
        if (content.Equals(""))
        {
            MessageBox.Show("No hay nada que mostrar escriba codiog para mostrar la web");
        }
        else
        {
            h = new Navegador(content);
            this.Invoke(new Action(() => { h.Refresh(); }));
            h.Titulo.Text = Titulo.Text;
            h.Show();

        }
    }
    else
    {
        if (fastColoredTextBox1.Language == FastColoredTextBoxNS.Language.HTML)
        {

            if (Application.OpenForms["Navegador"] == null && !fastColoredTextBox1.Text.Equals(""))
            {
                if (!h.IsDisposed)
                {
                    h.Actualizar(fastColoredTextBox1.Text);
                    this.Invoke(new Action(() => { h.Refresh(); }));
                    h.Titulo.Text = Titulo.Text;
                    h.Show();
                    h.FormClosed += Logout;
                }
                else
                {
                    h = new Navegador(fastColoredTextBox1.Text);
                    h.Titulo.Text = Titulo.Text;
                    h.Show();
                    this.Invoke(new Action(() => { h.Refresh(); }));
                    h.FormClosed += Logout;
                }
            }
            else
            {

                h.Actualizar(fastColoredTextBox1.Text);
                this.Invoke(new Action(() => { h.Refresh(); }));
                h.Titulo.Text = Titulo.Text;
                h.FormClosed += Logout;

            }
```

```
            }
            else
            {
                MessageBox.Show("NO SE PUEDE EJECUTAR");
            }
        }
    }
    private void Logout(object sender, FormClosedEventArgs e)
    {

        this.Focus();

    }

    private void aYUDAToolStripMenuItem1_Click(object sender, EventArgs e)
    {
        if (Application.OpenForms["Ayuda"] == null)
        {
            Ayuda MenuDeAyuda = new Ayuda();
            MenuDeAyuda.Show();
        }
    }

    private void AtrasoolStripMenuItem_Click(object sender, EventArgs e)
    {
        fastColoredTextBox1.Undo();
    }

    bool div = false;
    private void fastColoredTextBox1_KeyPress(object sender, KeyPressEventArgs e)
    {
        if (e.KeyChar.ToString() == Convert.ToString("/"))
        {
            div = true;
        }
        if (e.KeyChar.ToString() == Convert.ToString(">") && div == true)
        {
            div = false;
        }
    }
```

1.25 TMFECHAHORA:

```
        //METODO PARA HORA Y FECHA ACTUAL ------------------------------------------------------------
        private void tmFechaHora_Tick(object sender, EventArgs e)
        {
            lbFecha.Text = DateTime.Now.ToLongDateString();
            lblHora.Text = DateTime.Now.ToString("HH:mm:ssss");
        }




    }
}
```

2 DOM.CS

2.1 DOM:

```
public partial class DOM : Form
{
    string pal;
    string[] trozos1;
    public DOM()
    {
        InitializeComponent();
    }

    public void Graficar(int[] columna)
    {
        Graphics nodo;
        nodo = CreateGraphics();


        int tam = 580;

        int fila = columna.Length;

        int sum = columna.Sum();

        int k = 0;

        Font drawFont = new Font("Arial", 13);

        int y =30;
        int x = 50;
        int z = 0;
        int[] a = new int[50];//
        int[] b = new int[50];//
        Pen linea = new Pen(Color.Gold, 2);

        for (int i = 0; i < fila; i++)
        {
            x = (tam / (columna[i] + 1));
            z = x;
            for (int j = 0; j < columna[i]; j++)
            {

                nodo.FillRectangle(Brushes.White, x, y, 78, 35);
                nodo.DrawString(trozos1[k], drawFont, Brushes.Black, x + 13, y + 5);//PINTA EL TEXTO
                a[k] = x;//CUENTA PUNTOS PARA LINEAS
                b[k] = y;//CUENTA PUNTOS PARA LINEAS
                x += z;
                k++;//AYUDA A PINTAR EL TEXTO
            }

            y += 60;

        }
```

```
    int xi = 300;
    int yi = 60;
    int xf = xi;
    int yf = yi + 30;
    z = 0;
    int w = 0;

    for (int i = 1; i < fila; i++)//------
    {
        if (i != fila - 2)
        {
            xi = ((tam + 60) / (columna[i] + 1));
            z = xi;
            xf = xi;
            for (int j = 0; j < columna[i]; j++)
            {

                nodo.DrawLine(linea, xi, yi, xf, yf);
                xi += z;
                xf = xi;
                //xf += z;

            }
        }
        else
        {
            xf = ((tam + 60) / (columna[i] + 1));
            z = xf;
            xi = ((tam + 60) / (columna[i - 1] + 1));
            for (int j = 0; j < columna[i]; j++)
            {

                nodo.DrawLine(linea, xi, yi + 5, xf, yf);
                xf += z;



            }
        }
        yi = (yf + 30);
        yf = (yi + 30);


    }



}//TERMINA LA FUNCION
```

2.2 ARBOL:

```
static string arbol(string pal)
{
    string ncadena = "";
    int ln = (pal).Length;//*************
    int parentesis1 = 0, parentesis2 = 0, inicio = 0, tamaño = 0, copia = 0;
    string nodo = "", cnodo = "", hijo = "";


    parentesis1 = pal.IndexOf("/");
    parentesis2 = pal.IndexOf(">", parentesis1);//****
    int resta = parentesis2 - parentesis1;
    nodo = pal.Substring(parentesis1 + 1, resta - 1);
    cnodo = nodo;
    nodo = "<" + nodo + ">";


    inicio = pal.IndexOf(nodo);
    tamaño = (nodo).Length;//LONGITUD DEL NOD

    copia = inicio + tamaño;
    resta = (parentesis1 - 1) - copia;

    hijo = pal.Substring(copia, resta);//DONE ESTA EL HIJO

    ncadena = pal.Substring(0, inicio) + cnodo + "(" + hijo + ")";
    resta = ln - (parentesis2 + 1);
    ncadena += pal.Substring(parentesis2 + 1, resta);

    return ncadena;

}
```

2.3 PROCESO:

```
public int[] proceso(string array)
{

    int tamaño = (array).Length;
    int fila = 0, j = 0, aux = 0, aux1 = 0, profi = 0;
    int i = 0, p = 0;
    char[] arbol = array.ToCharArray();

    int[] c = new int[20];
    char[] imprimiraux = new char[500];
    int[] imporden = new int[500];

    foreach (char ch in arbol)
    {

        if (ch == '(')
        {
            aux1 = fila;
            fila++;

            if (aux1 < fila)
            {

                c[i] = fila;
                i++;
            }

            aux1 = fila;



        }
        if (ch == ')')
        {
            fila -= 1;
            aux1 = fila;
            imprimiraux[p] = '+';///
            imporden[p] = aux1;///
            p++;///



        }
        if (ch != '(' && ch != ')' && ch != ',' && ch != '.' && ch != ' ')
        {
            j++;
            imprimiraux[p] = ch;
            imporden[p] = aux1;
            p++;

        }
```

```
int max = 0;
for (int k = 0; k < c.Length; k++)
{
    if (c[k] > max)
    {
        max = c[k];
    }

}

profi = max + 1;
//Console.WriteLine("PROFUNDIDAD: " + profi);
string[] primera = new string[50];
string juntar = "";

for (int x = 0; x < profi; x++)
{

    int variables = (imporden).Length;

    //Console.Write( x.ToString()+"|");
    for (int y = 0; y < variables; y++)
    {
        if (imporden[y] == x)
        {

            //Console.Write(imprimiraux[y].ToString());
            juntar += imprimiraux[y].ToString();

        }


    }
    primera[x] = juntar;
    juntar = "";
    //Console.WriteLine();
}


primera[profi - 1] = proceso2(array);
for (int t = 0; t < profi; t++)
{
    Console.WriteLine(primera[t]);
}
```

```
int count = 0;
int[] dev = new int[profi];
for (int t = 0; t < profi; t++)
{
    count = primera[t].Split('+').Length - 1;
    dev[t] = count;
    //Console.WriteLine(count);

}

//////////////////////////////////////////////////////////////////7

string signo = primera[profi - 1];
signo = signo.Substring(1);
primera[profi - 1] = signo;

string codigo = "";

for (int t = 0; t < profi; t++)
{

    codigo += primera[t];////

}
//Console.WriteLine(codigo.Trim());
char[] delimitador = { '+', ' ' };
trozos1 = codigo.Split(delimitador);
//trozos1 = trozos;


return dev;
```

2.4 PROCESO2:

```
static string proceso2(string array)//22222222222222222222222222222222222222222222222222222222222222222
{

    int tamaño = (array).Length;
    int fila = 0, j = 0, aux = 0, aux1 = 0, profi = 0;
    int i = 0, p = 0;
    char[] arbol = array.ToCharArray();

    int[] c = new int[20];
    char[] imprimiraux = new char[500];
    int[] imporden = new int[500];

    foreach (char ch in arbol)
    {

        if (ch == '(')
        {
            aux1 = fila;
            fila++;

            if (aux1 < fila)
            {

                c[i] = fila;
                i++;
            }

            aux1 = fila;

            imprimiraux[p] = '+';///
            imporden[p] = aux1;///
            p++;///

        }
        if (ch == ')')
        {
            fila -= 1;
            aux1 = fila;




        }
        if (ch != '(' && ch != ')' && ch != ',' && ch != '.' && ch != ' ')
        {
            j++;
            imprimiraux[p] = ch;
            imporden[p] = aux1;
            p++;
```

```
    int max = 0;
    for (int k = 0; k < c.Length; k++)
    {
        if (c[k] > max)
        {
            max = c[k];
        }

    }

    profi = max + 1;
    // Console.WriteLine("PROFUNDIDAD: " + profi);
    string[] primera = new string[50];
    string juntar = "";

    for (int x = 0; x < profi; x++)
    {

        int variables = (imporden).Length;

        //Console.Write( x.ToString()+"|");
        for (int y = 0; y < variables; y++)
        {
            if (imporden[y] == x)
            {

                //Console.Write(imprimiraux[y].ToString());
                juntar += imprimiraux[y].ToString();

            }


        }
        primera[x] = juntar;
        juntar = "";
        //Console.WriteLine();
    }
    for (int t = 0; t < profi; t++)
    {
        //Console.WriteLine(primera[t]);
    }
    // Console.WriteLine(primera[profi - 1]);
    // int count = primera[profi - 1].Split('+').Length - 1;
    // Console.WriteLine(count);
    return primera[profi - 1];

}//TERMINA LA FUNCION
```

2.5 DOM_ACTIVATED_1:

```
        private void DOM_Activated_1(object sender, EventArgs e)
        {
            pal = FormMenuPrincipal.variableCom;
            if (pal.Length > 5)
            {
                pal = pal.Replace("\n", "");
                pal = pal.Replace(" ", "");
                string regresa = "";
                bool i = true;
                while (i == true)
                {
                    regresa = arbol(pal);
                    pal = regresa;
                    i = pal.Contains("/");

                }

                int[] resivir = (proceso(pal));
                Graficar(resivir);
            }
        }

        private void DOM_Shown(object sender, EventArgs e)
        {
            pal = FormMenuPrincipal.variableCom;
            pal = pal.Replace("\n", "");
            pal = pal.Replace(" ", "");
            string regresa = "";
            bool i = true;


            while (i == true)
            {
                regresa = arbol(pal);
                pal = regresa;
                i = pal.Contains("/");

            }

            int[] resivir = (proceso(pal));
            Graficar(resivir);

        }




    }
}
```

3 AYUDA.CS

3.1 AYUDA:

```
public Ayuda()
{
    InitializeComponent();
    //Estas lineas eliminan los parpadeos del formulario o controles en la interfaz grafica (Pero no en un 100%)
    this.SetStyle(ControlStyles.ResizeRedraw, true);
    this.DoubleBuffered = true;

}
//METODO PARA REDIMENCIONAR/CAMBIAR TAMAÑO A FORMULARIO  TIEMPO DE EJECUCION ----------------------------------------
private int tolerance = 15;
private const int WM_NCHITTEST = 135;
private const int HTBOTTOMRIGHT = 17;
private Rectangle sizeGripRectangle;

protected override void WndProc(ref Message m)
{
    switch (m.Msg)
    {
        case WM_NCHITTEST:
            base.WndProc(ref m);
            var hitPoint = this.PointToClient(new Point(m.LParam.ToInt32() & 0xffff, m.LParam.ToInt32() >> 16));
            if (sizeGripRectangle.Contains(hitPoint))
                m.Result = new IntPtr(HTBOTTOMRIGHT);
            break;
        default:
            base.WndProc(ref m);
            break;
    }
}

//METODO PARA ARRASTRAR EL FORMULARIO----------------------------------------------------------------------------
[DllImport("user32.DLL", EntryPoint = "ReleaseCapture")]
private extern static void ReleaseCapture();

[DllImport("user32.DLL", EntryPoint = "SendMessage")]
private extern static void SendMessage(System.IntPtr hWnd, int wMsg, int wParam, int lParam);

private void PanelBarraTitulo_MouseDown(object sender, MouseEventArgs e)
{
    ReleaseCapture();
    SendMessage(this.Handle, 0x112, 0xf012, 0);
}
```

```
//METODOS PARA CERRAR,MAXIMIZAR, MINIMIZAR FORMULARIO-----------------------------------------------------
int lx, ly;
int sw, sh;
private void btnMaximizar_Click(object sender, EventArgs e)
{
    lx = this.Location.X;
    ly = this.Location.Y;
    sw = this.Size.Width;
    sh = this.Size.Height;
    this.Size = Screen.PrimaryScreen.WorkingArea.Size;
    axAcroPDF1.Size = Screen.PrimaryScreen.WorkingArea.Size;
    this.Location = Screen.PrimaryScreen.WorkingArea.Location;
    btnMaximizar.Visible = false;
    btnNormal.Visible = true;

}

private void btnNormal_Click(object sender, EventArgs e)
{
    this.Size = new Size(sw, sh);
    axAcroPDF1.Size = new Size(sw, sh);
    this.Location = new Point(lx, ly);
    btnNormal.Visible = false;
    btnMaximizar.Visible = true;
}

private void button1_Click(object sender, EventArgs e)
{

    axAcroPDF1.src = Path.GetDirectoryName(Path.GetDirectoryName(System.IO.Directory.GetCurrentDirectory())) + @"\Resources\Manual De Usuario.pdf";
}

private void button2_Click(object sender, EventArgs e)
{
    axAcroPDF1.src = Path.GetDirectoryName(Path.GetDirectoryName(System.IO.Directory.GetCurrentDirectory())) + @"\Resources\Manual De Desarrollador.pdf";
}

private void button3_Click(object sender, EventArgs e)
{
    //axAcroPDF1.src = Path.GetDirectoryName(Path.GetDirectoryName(System.IO.Directory.GetCurrentDirectory())) + @"\Resources\Creditos.pdf";
    MessageBox.Show("RV\nEG\nJV");
}

private void btnMinimizar_Click(object sender, EventArgs e)
{
    this.WindowState = FormWindowState.Minimized;
}

private void btnCerrar_Click(object sender, EventArgs e)
{
    this.Close();
}
```

4 NAVEGADOR.CS

4.1 NAVEGADOR

```
public Navegador(string file)
{
    InitializeComponent();
    //Estas lineas eliminan los parpadeos del formulario o controles en la interfaz grafica (Pero no en un 100%)
    this.SetStyle(ControlStyles.ResizeRedraw, true);
    this.DoubleBuffered = true;
    webBrowser1.DocumentText = file;
    Titulo.Text = "Sin titulo";

}
//METODO PARA REDIMENCIONAR/CAMBIAR TAMAÑO A FORMULARIO  TIEMPO DE EJECUCION ------------------------------------------------------------
private int tolerance = 15;
private const int WM_NCHITTEST = 135;
private const int HTBOTTOMRIGHT = 17;
private Rectangle sizeGripRectangle;

protected override void WndProc(ref Message m)
{
    switch (m.Msg)
    {
        case WM_NCHITTEST:
            base.WndProc(ref m);
            var hitPoint = this.PointToClient(new Point(m.LParam.ToInt32() & 0xffff, m.LParam.ToInt32() >> 16));
            if (sizeGripRectangle.Contains(hitPoint))
                m.Result = new IntPtr(HTBOTTOMRIGHT);
            break;
        default:
            base.WndProc(ref m);
            break;
    }
}
//----------------DIBUJAR RECTANGULO / EXCLUIR ESQUINA PANEL
protected override void OnSizeChanged(EventArgs e)
{
    base.OnSizeChanged(e);
    var region = new Region(new Rectangle(0, 0, this.ClientRectangle.Width, this.ClientRectangle.Height));

    sizeGripRectangle = new Rectangle(this.ClientRectangle.Width - tolerance, this.ClientRectangle.Height - tolerance, tolerance, tolerance);

    region.Exclude(sizeGripRectangle);
    this.panelContenedorPrincipal.Region = region;
    this.Invalidate();
}
```

4.2 OnPaint

```
protected override void OnPaint(PaintEventArgs e)
{

    SolidBrush blueBrush = new SolidBrush(Color.FromArgb(55, 61, 69));
    e.Graphics.FillRectangle(blueBrush, sizeGripRectangle);

    base.OnPaint(e);
    ControlPaint.DrawSizeGrip(e.Graphics, Color.Transparent, sizeGripRectangle);
}

//METODO PARA ARRASTRAR EL FORMULARIO----------------------------------------------------------------------------
[DllImport("user32.DLL", EntryPoint = "ReleaseCapture")]
private extern static void ReleaseCapture();

[DllImport("user32.DLL", EntryPoint = "SendMessage")]
private extern static void SendMessage(System.IntPtr hWnd, int wMsg, int wParam, int lParam);

private void PanelBarraTitulo_MouseDown(object sender, MouseEventArgs e)
{
    ReleaseCapture();
    SendMessage(this.Handle, 0x112, 0xf012, 0);
}
//METODOS PARA CERRAR,MAXIMIZAR, MINIMIZAR FORMULARIO-----------------------------------------------------------
int lx, ly;
int sw, sh;
private void btnMaximizar_Click(object sender, EventArgs e)
{
    lx = this.Location.X;
    ly = this.Location.Y;
    sw = this.Size.Width;
    sh = this.Size.Height;
    this.Size = Screen.PrimaryScreen.WorkingArea.Size;
    this.Location = Screen.PrimaryScreen.WorkingArea.Location;
    btnMaximizar.Visible = false;
    btnNormal.Visible = true;

}
```

4.3 BTNNORMAL_CLICK

.

```
private void btnNormal_Click(object sender, EventArgs e)
{
    this.Size = new Size(sw, sh);
    this.Location = new Point(lx, ly);
    btnNormal.Visible = false;
    btnMaximizar.Visible = true;
}

private void btnMinimizar_Click(object sender, EventArgs e)
{
    this.WindowState = FormWindowState.Minimized;
}

private void btnCerrar_Click(object sender, EventArgs e)
{
    this.Close();
}

public void Actualizar(string file)
{
    webBrowser1.DocumentText = file;

}
```