

1. Gráficos.

1.1 Introducción.

En Vb Net como mínimo hay que aproximarse al apartado gráficos, pues en el uso de la impresora es necesario el uso de varios de esos elementos, por lo tanto al menos hay que hacer una descripción de los mismos.

El uso de las fuentes y los dibujos en Studio Net igual que otros conceptos han sido modificados.

Antes en Vb 6, se podía cambiar las características de las fuentes, o de los gráficos de cualquier objeto en tiempo de ejecución, ahora también, pero la sintaxis y la forma han cambiado bastante.

Con el color sucede lo mismo, el color se asigna bajo las mismas circunstancias.

Para dibujar líneas, disponemos de las características de Brush, donde asignamos colores, anchos.

Igual sucede con la ubicación actual del punto de inserción, la propiedad CurrentX y CurrentY, ahora van como parámetros en el método de la clase Drawing.

En Studio Net aparece un objeto denominado Graphics, que es el área donde se va a realizar el dibujo con uno de los métodos de dibujo.

Dicho apartado podríamos llamarlo algo así como dibuja, pinta y escribe.

¿Por qué?, porque los elementos que intervienen en la escritura en la impresora son

Pen
Brush
Font
Color

1.2 System.Drawing

Es un espacio de nombres que alberga distintos elementos, entre ellos Drawing.

System.Drawing, proporciona medios para dibujar.

Describir todos los métodos y demás elementos no tiene sentido, así que lo mejor es pinchar en el link de Microsoft y allí ver todas las posibilidades de cada uno de los métodos y propiedades.

[http://msdn2.microsoft.com/es-es/system.drawing\(VS.80\).aspx](http://msdn2.microsoft.com/es-es/system.drawing(VS.80).aspx)

1.3 Graphics.

En Studio Net se parte del criterio que para escribir o dibujar algo primero hay que crear algo donde hacerlo, ese algo es un objeto Graphics..

Por el mero hecho de colocar un objeto en el formulario, dicho objeto, a parte de que puede ser o no capaz de recibir un texto o imagen, no dispone de la capacidad de recibir un gráfico, por agrupar el concepto de dibujo y escritura.

Por lo tanto, lo que se hace es que al objeto del que deseamos obtener la prestación de crear un gráfico, le asociamos un objeto gráfico, el cual será capaz de recibir esas prestaciones.

Así que para el uso de un objeto Graphics es necesario primero su creación.

Como antes hemos comentado, éste objeto será un área de dibujo, y como tal se ha de corresponder con algún objeto, por ejemplo el formulario actual, por lo tanto crear un objeto Graphics podría quedar así:

```
` Crear un elemento gráfico con el formulario actual.  
Dim Grafico as Graphics = Me.CreateGraphics
```

En los ejemplos de Microsoft, en MSDN, aparece con relativa frecuencia la siguiente línea de código, o equivalente,

```
NombreProcedimiento (ByVal e As PaintEventArgs)  
e.Graphics.DrawString(Texto, Fuente, Pincel.Black, x,y, formato)
```

El objeto *e* se puede sustituir por la línea

```
` Crear un elemento gráfico con el formulario actual.  
Dim Grafico as Graphics = Me.CreateGraphics
```

dentro del procedimiento, y sustituir por *grafico* el objeto *e*, u otro nombre claro está, con lo que podría quedar así.

```
NombreProcedimiento ()  
` Crear un elemento gráfico con el formulario actual.  
Dim Grafico as Graphics = Me.CreateGraphics  
  
Grafico.DrawString(Texto, Fuente, Pincel.Black, x,y,formato)
```

Solo tiene sentido para los ejemplos que podamos ver mientras aprendemos el funcionamiento de .Net. Otra posibilidad es que en el procedimiento se haga el envío solamente del objeto *e.graphics*, con lo que solo hay que preocuparse de recibir en nuestro procedimiento un objeto *graphics*,

```
Dibuja(e.Graphics)  
  
Private Sub Dibuja(ByVal Grafico As Graphics)  
  
End Sub
```

Una vez creado el elemento gráfico, ya se podría crear un dibujo en el mismo, y sería suficiente utilizar uno de sus métodos gráficos,

```
Grafico.DrawEllipse(Lapiz, 20, 30, 10, 50)
```

por lo que el resultado sería ...

```
` Crear un elemento gráfico en el formulario actual.  
Dim Grafico as Graphics = Me.CreateGraphics  
Grafico.DrawEllipse(Lapiz, 20, 30, 10, 50)
```

Una vez creado el objeto *Graphics*, éste se puede usar para dibujar líneas y formas, representar texto o mostrar y manipular imágenes.

```
Public Class Form1  
Private Sub Dibujar()  
` Crear el lápiz  
Dim Lapiz As New Pen(Color.Red)  
` Crear un elemento gráfico con el formulario actual.  
Dim Grafico As Graphics = Me.CreateGraphics  
` Dibuja la elipse  
Grafico.DrawEllipse(Lapiz, 20, 30, 10, 50)  
` Liberar recursos  
Grafico.Dispose()  
End Sub
```

```

Private Sub Form1_Click(ByVal sender As Object, _
                        ByVal e As System.EventArgs) Handles Me.Click
    Dibujar()
End Sub
End Class

```

En el ejemplo anterior se dibuja una elipse de color rojo, que es el color asignado al objeto lápiz.

Los objetos principales que se usan con el objeto Graphics son

- El lápiz, Pen
- El pincel, Brush
- La fuente. Font
- El color, Color.
- Formatos, StringFormat

Los objetos lápiz y pincel se usan para representar gráficos, texto e imágenes con la interfaz GDI+.

Un lápiz es una instancia de la clase Pen, y se usa para dibujar líneas y contornos de formas.

Un pincel es una instancia de cualquier clase que se derive de la clase Brush MustInherit (abstract), y se puede usar para rellenar formas o dibujar texto.

También se puede crear un objeto gráfico a partir de un archivo de imagen, como se puede ver en el ejemplo que sigue.

```

Dim MiMapaBits as New Bitmap("C:\MiArchivo.Bmp")
Dim Grafico as Graphics = Graphics.FromImage(MiMapaBits)

```

1.4 Lápiz, Pen, Clase.

Pertenece al espacio de nombres System.Drawing.

Se usa como elemento dentro del método adecuado para realizar un dibujo, como líneas, curvas y el contorno de las formas, u otros elementos geométricos.

El ejemplo siguiente crea un lápiz de color negro.

```

` Crear el lápiz
Dim Lapis as New Pen(Color.Red)
` Crear un pincel.
Dim Pincel as New SolidBrush(Color.Blue)

```

Cuando ya se ha creado el lápiz, se puede utilizar.

```

` Crear el lápiz
Dim Lapis as New Pen(Color.Black)
` Crear un elemento gráfico con el formulario actual.
Dim Grafico as Graphics = Me.CreateGraphics
Grafico.DrawEllipse(Lapis, 20, 30, 10, 50)

```

Con lo anterior creado ya se puede dibujar una elipse como figura en el ejemplo, y el resultado final puede ser:

```
Private Sub Dibujar
    ` Crear el lápiz
    Dim Lápiz as New Pen(Color.Black)
    ` Crear un elemento gráfico con el formulario actual.
    Dim Grafico as Graphics = Me.CreateGraphics
    ` Dibuja la elipse
    Grafico.DrawEllipse(Lápiz, 20, 30, 10, 50)
    ` Liberar recursos
    Lápiz.Dispose
    Grafico.Dispose
End Sub
```

Conviene después del uso de estos objetos utilizar el método Dispose para liberar recursos.

Para poder acceder a todas sus características podemos acceder al link de Microsoft:

[http://msdn2.microsoft.com/es-es/library/system.drawing.pen_members\(VS.80\).aspx](http://msdn2.microsoft.com/es-es/library/system.drawing.pen_members(VS.80).aspx)

1.5 Pincel, Brush.

Esta clase es utilizada para rellenar gráficos, rectángulos, elipses, círculos.

En realidad es como si se defieran las condiciones con las que se va a realizar un dibujo, en el mismo momento en el que éste se ejecuta.

En la versión anterior de VB, antes de utilizar el método Line, se definían las características del tipo de traza con el que se iba a realizar, el espesor y el color

Los pinceles son objetos que se usan con un objeto Graphics para crear formas sólidas y para representar texto. Existen varios tipos distintos de pinceles:

Tipo de pincel	Descripción
SolidBrush	La forma más simple de pincel, que pinta en un color sólido.
HatchBrush	Similar a SolidBrush, pero permite seleccionar de entre una amplia variedad de modelos preestablecidos para dibujar, en lugar de usar un color sólido.
TextureBrush	Dibuja con una textura, por ejemplo una imagen.
LinearGradientBrush	Dibuja dos colores mezclados a lo largo de un gradiente.
PathGradientBrush	Dibuja con un gradiente complejo de colores mezclados, basado en una ruta única definida por el programador.

Todas estas clases heredan de la clase Brush, que es una clase abstract (MustInherit) y de la que, por tanto, no se pueden crear instancias.

En el ejemplo que sigue, creamos un pincel, un gráfico y luego dibujamos en él una elipse.

```
Private Sub DibujarElipse()
    Dim Pincel As New _
        System.Drawing.SolidBrush(System.Drawing.Color.LemonChiffon)
    Dim Dibujo As System.Drawing.Graphics
    Dibujo = Me.CreateGraphics()
    Dibujo.FillEllipse(Pincel, New Rectangle(0, 0, 100, 200))
    Pincel.Dispose()
End Sub
```

Para poder acceder a todas sus características podemos acceder a éste link:

[http://msdn2.microsoft.com/es-es/library/system.drawing.pen_members\(VS.80\).aspx](http://msdn2.microsoft.com/es-es/library/system.drawing.pen_members(VS.80).aspx)

1.6 Color, Color.

Los objetos Color son instancias de clases que representan a un color determinado, y tanto los lápices como los pinceles los pueden usar para indicar el color de los gráficos representados.

Color es una estructura que pertenece al espacio de nombres System.Drawing.

Color es una colección de colores predefinidos con nombre para poder utilizar en cualquier momento, en el que los colores están definidos como propiedad.

Es una colección muy amplia, y es mejor recurrir a la ayuda y probar los colores disponibles.

```
Dim Pincel As System.Drawing.SolidBrush
Pincel = New System.Drawing.SolidBrush(System.Drawing.Color.PeachPuff)
```

Después de la definición del objeto, solo queda utilizarlo con el color con el que se ha definido.

```
Private Sub DibujarElipse()
    Dim Pincel As System.Drawing.SolidBrush
    Pincel = New System.Drawing.SolidBrush(System.Drawing.Color.PeachPuff)
    Dim Dibujo As System.Drawing.Graphics
    Dibujo = Me.CreateGraphics()
    Dibujo.FillEllipse(Pincel, New Rectangle(0, 0, 100, 200))
    Pincel.Dispose()
End Sub
```

También puede hacerse la definición como en ejemplos anteriores.

```
Dim Pincel As New _
    System.Drawing.SolidBrush(System.Drawing.Color.Red)
```

Para poder acceder a todas sus características podemos acceder a éste link:

[http://msdn2.microsoft.com/es-es/system.drawing.color_members\(VS.80\).aspx](http://msdn2.microsoft.com/es-es/system.drawing.color_members(VS.80).aspx)

1.7 Fuente, Font, Clase.

Pertenece al espacio de nombres System.Drawing.

Ahora no es posible cambiar las propiedades de las fuentes de un objeto, y los cambios de fuente están basados en las herencias, de tal forma que si se cambia las características de la fuente de un objeto, los objetos que de él han sido heredados cambiarán también.

Si en un formulario tenemos una colección de objetos, que han heredado las fuentes de éste, entonces cuando sean modificadas las fuentes del formulario éstos también modificarán sus fuentes.

La clase Font, define un formato concreto para el texto, incluidos el nombre de fuente, el tamaño y los atributos de estilo, pertenece al espacio de nombres System.Drawing.

Si deseamos saber cuales son las fuentes disponibles en nuestro sistema podemos ejecutar el siguiente código, en un formulario con un ListBox llamado Lista.

```
Private Sub VerFuentes()
    Dim Fuente As FontFamily
    For Each Fuente In System.Drawing.FontFamily.Families
        Lista.Items.Add(Fuente.Name)
    Next
End Sub
```

En el siguiente ejemplo se muestra como cargar el cuadro de dialogo de las fuentes con los parámetros por fuente del objeto TextBox.

Después al objeto `TextBox` le hemos cambiado sus características y volvemos a asignarlas al cuadro de dialogo de las fuentes, y podremos observar que los valores por defecto del mismo en su inicio son distintos.

```
Private Sub VerEstiloDos()  
  
    FontDialog1.ShowColor = True  
    FontDialog1.Font = TextBox1.Font  
    FontDialog1.Color = TextBox1.ForeColor  
  
    If FontDialog1.ShowDialog() <> DialogResult.Cancel Then  
        TextBox1.Font = FontDialog1.Font  
        TextBox1.ForeColor = FontDialog1.Color  
    End If  
  
    TextBox1.Font = New Font("Arial", 10, FontStyle.Underline)  
  
    FontDialog1.ShowColor = True  
    FontDialog1.Font = TextBox1.Font  
    FontDialog1.Color = TextBox1.ForeColor  
  
    If FontDialog1.ShowDialog() <> DialogResult.Cancel Then  
        TextBox1.Font = FontDialog1.Font  
        TextBox1.ForeColor = FontDialog1.Color  
    End If  
  
End Sub
```

El uso completo de la escritura de texto esta basado en un color, en una fuente a utilizar en un objeto determinado. Por lo tanto el uso completo puede ser el que sigue.

```
Private Sub EscribirTexto()  
    Dim Grafico As System.Drawing.Graphics = Me.CreateGraphics()  
    Dim Texto As String = "Texto a escribir"  
    Dim Fuente As New System.Drawing.Font("Arial", 16)  
    Dim Pincel As New _  
  
    System.Drawing.SolidBrush(System.Drawing.Color.LightCyan)  
    Dim x As Single = 650.0  
    Dim y As Single = 500.0  
    Dim Formato As New System.Drawing.StringFormat  
  
    Grafico.DrawString(Texto, Fuente, Pincel, x, y, Formato)  
    Fuente.Dispose()  
    Pincel.Dispose()  
    Grafico.Dispose()  
  
End Sub
```

Conviene después del uso de estos objetos utilizar el método `Dispose` para liberar recursos.

A continuación se visualizan las propiedades por así decirlo clásicas o ya conocidas

1.7.1 Bold.

Obtiene la información para saber si la fuente esta en negrita, o no.
Su valor es `true` o `false`.

1.7.2 Height.

Devuelve el interlínea de esa fuente.

1.7.3 Italic.

Devuelve si está en cursiva o no la fuente actual.

1.7.4 Name

Devuelve el nombre de la fuente actual.

1.7.5 Size.

Devuelve el tamaño usado de la fuente actual.

1.7.6 StrikeOut

Indica si está o no activado el tachado.

1.7.7 UnderLine

Indica si está o no activado el subrayado.

1.7.8 FontFamily

Devuelve la colección de fuentes que están asociadas al objeto Font actual.

1.7.9 GDIVerticalFont

Devuelve true o false, en función de que se derive de una fuente que admita la orientación vertical.

1.7.10 Style

Obtiene la información del estilo del objeto.

1.7.11 SizeInPoints

Devuelve el tamaño usado de la fuente actual, expresado en puntos.

Para más información sobre Font, acceder mediante éste link

[http://msdn2.microsoft.com/es-es/library/system.drawing.font_members\(VS.80\).aspx](http://msdn2.microsoft.com/es-es/library/system.drawing.font_members(VS.80).aspx)

1.8 StringFormat, Formato.

Para completar el aspecto referente a la visualización del texto, falta el aspecto de su representación referente al como representar los datos, interlinea y alineación.

Encapsula información de diseño del texto (como interlineado y alineación), manipulaciones de presentación (como inserción de puntos suspensivos y sustitución de dígitos nacional) y características de OpenType.

Con esta clase obtenemos la posibilidad de crear formatos en el momento de la visualización de la información.

El formato se obtiene mediante la combinación de los parámetros de

Alignement
DigitSubstitutionLanguage
DigitSubstitutionMethod
FormatFlags.
LineAlignement
Trimming

La información de éste apartado se puede consultar en el siguiente link;

[http://msdn2.microsoft.com/es-es/library/system.drawing.stringformat_members\(VS.80\).aspx](http://msdn2.microsoft.com/es-es/library/system.drawing.stringformat_members(VS.80).aspx)

1.8.1 Alignement.

Obtiene o establece la alineación del texto con respecto al plano vertical.

Utiliza la enumeración StringAlignement.

Sus valores son :

Center,
Far,

Near,

Texto centrado.

Alejado del inicio del sentido de escritura, lo que significa que en nuestra cultura alineación derecha.

Cercano al inicio del sentido de escritura, lo que significa en nuestra cultura alineación izquierda.

Lo de alejado y cercano viene dado por la posibilidad de que el sentido de escritura sea de derecha a izquierda propio de la cultura árabe, pero no de la occidental.

1.8.2 LineAlignment.

Obtiene o establece la alineación del texto con respecto al plano horizontal
Utiliza la misma enumeración que Alignment, StringAlignment.

1.8.3 DigitSubstitutionLanguage.

Obtiene o establece el idioma que se utiliza cuando los dígitos locales se sustituyen por dígitos occidentales.

1.8.4 DigitSubstitutionMethod

Obtiene o establece el método que se va a utilizar para la sustitución de dígitos.
Utiliza la enumeración StringDigitSubstitute
Sus valores son

None	Que se deshabilitan las sustituciones.
National.	Según el idioma nacional oficial de la configuración regional del usuario.
Traditional	Se corresponderá con el alfabeto o el idioma nativos del usuario, que pueden no coincidir con el idioma nacional oficial de la configuración regional del usuario.
User	Se utiliza un esquema definido por el usuario

1.8.5 FormatFlags.

Especifica la información de diseño y de presentación de las cadenas de texto.
Dispone de diversos valores que permiten definir como se visualizará el texto a representar.
Utiliza la enumeración SStringFormatFlags.
Sus valores más son:

Orientación vertical	DirectionVertical.
Sentido desde la derecha o la izquierda,	DirectionRightToLeft

Como visualizar el texto dentro y en los límites de la zona asignada al mismo.
La reserva de fuentes alternativas para los caracteres que no admita la fuente solicitada queda deshabilitada. Los caracteres que falten se muestran con el glifo de ausencia de fuente, normalmente un cuadrado abierto. Utilización de las fuentes, NoFontFallback.

Acudir a esta dirección para su consulta:

<http://msdn.microsoft.com/library/spa/default.asp?url=/library/SPA/cpref/html/frlrfsystemdrawingstringformatflagsclasstopic.asp>

1.8.6 Trimming.

Indica como recortar una cadena que no cabe en el área asignada al objeto.
Utiliza la enumeración StringTrimming.

Sus valores son:

Character.	Se recorta hasta el carácter más cercano.
EllipsisCharacter.	Se recorta hasta el carácter más cercano y se insertan puntos suspensivos al final de la línea recortada.
EllipsisPath	En las líneas recortadas se elimina el centro y se sustituye por puntos suspensivos. El algoritmo mantiene la mayor parte posible del último segmento delimitado por barra diagonal de la línea.

EllipsisWord	Especifica que el texto se recorta hasta la palabra más cercana y se insertan puntos suspensivos al final de la línea recortada.
None	No se hace el recorte
Word	Se hace hasta la palabra más cercana.

El ejemplo que sigue muestra como se puede asignar formato al texto que se escribe en un objeto gráfico.

Para su prueba solo es necesario llamarlo desde cualquier evento en un formulario.

```

Private Sub EscribirTexto()
    ' Definición de objetos
    Dim Grafico As System.Drawing.Graphics = Me.CreateGraphics()
    Dim Texto As String = "Texto ejemplo"
    Dim Fuente As New System.Drawing.Font("Arial", 16)
    Dim Pincel As System.Drawing.SolidBrush(System.Drawing.Color.Black)
    Dim Formato As New System.Drawing.StringFormat
    ' Coordenadas
    Dim x As Single = 40.0
    Dim y As Single = 25.0

    ' Formato del texto
    Formato.FormatFlags = StringFormatFlags.DirectionVertical
    Formato.LineAlignment = StringAlignment.Far
    Formato.Trimming = StringTrimming.EllipsisCharacter

    ' ;;; Dibujado del texto!!!!
    Grafico.DrawString(Formato.Trimming.ToString, Fuente, Pincel, x, y,
Formato)
    ' Formato horizontal de derecha a izquierda
    x = Me.Width * 0.65 ' extremo derecho desde el que se escribe
    Formato.FormatFlags = StringFormatFlags.DisplayFormatControl
    Formato.Alignment = StringAlignment.Far
    ' Dibujado del texto
    Grafico.DrawString(Formato.FormatFlags.ToString, Fuente, Pincel, x,
y, Formato)
    ' las mismas coordenadas, con alineación y cambiando el sentido de
escritura
    Grafico.DrawString(Texto, Fuente, Pincel, Me.Width / 2, Me.Height /
2, Formato)
    Formato.LineAlignment = StringAlignment.Far
    Formato.FormatFlags = StringFormatFlags.DirectionRightToLeft
    Grafico.DrawString(Texto, Fuente, Pincel, Me.Width / 2, Me.Height /
2, Formato)

    ' Liberar recursos
    Fuente.Dispose()
    Pincel.Dispose()
    Grafico.Dispose()
End Sub

```