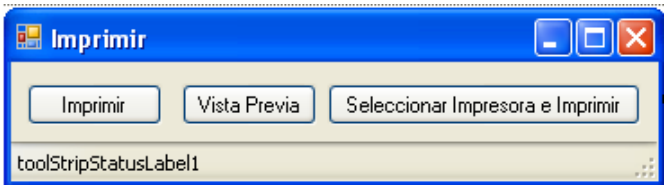
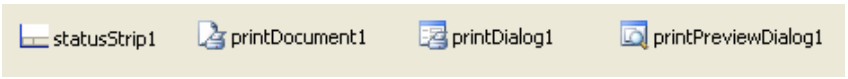
PROYECTO Imprimir:

Crear un Nuevo Proyecto desde Visual C# 2005 Express Edition, del tipo Aplicación de Escritorio para Windows, con el nombre Imprimir.





Interfaz visual: 1 formulario (Form1), 4 botones (button1, button2, button3 y button4) con los textos mostrados en el formulario adjunto, una barra de estado (statusStrip1) con una etiqueta de estado (toolStripStatusLabel1), y 3 controles relativos a la impresión (que al insertarlos desde el diseñador aparecen en la *zona inferior* del formulario):

1. Control printDocument (printDocument1) que permite imprimir un documento.
2. Control printDialog (printDialog1) que muestra el cuadro de diálogo común de selección de la impresora a usar y además permite imprimir.
3. Control printPreviewDialog (printPreviewDialog1) que permite mostrar el documento a imprimir en una ventana de diálogo de vista previa.

Para más información consultar esto: "Funcionalidad para imprimir en formularios Windows Forms": http://msdn.microsoft.com/library/spa/default.asp?url=/library/SPA/vbcon/html/vbtskprintinggraphics.asp

RECORDATORIO DEL FUNCIONAMIENTO BÁSICO DE LA IMPRESIÓN:

Ejemplo sencillo que muestra cómo construir un documento DE UNA ÚNICA PÁGINA que contiene un texto:

```
private void printDocument1_PrintPage(object sender,
    System.Drawing.Printing.PrintPageEventArgs e)
{
    string mensajeTexto = "Ejemplo de texto"; // Mensaje a mostrar.
    Font fuente = new Font("Arial", 12, FontStyle.Bold); // Fuente a usar.
    Brush brocha = Brushes.Black; // Color del texto.
    // Dibujamos el mensaje en negro en las coordenadas 150,125.
    e.Graphics.DrawString(mensajeTexto, fuente, brocha, 150, 125);
}
private void button1_Click(object sender, EventArgs e)
{
    printDocument1.Print(); // Imprimir el documento que crea printDocument1_PrintPage.
}
```

En **printDocument1_PrintPage** se coloca el código que construye el documento a imprimir, añadiendo los textos con formato (tipo de letra, tamaño, estilo, ...), formas gráficas (rectángulos, elipses, etc...) e imágenes, en coordenadas concretas de la página (cada uno en su lugar de forma precisa), mediante **e.Graphics**.

Desde otra parte del programa se ejecuta **printDocument1.Print**(), EN ESE MOMENTO SE EJECUTARÁ EL CÓDIGO DE LA FUNCIÓN **printDocument1_PrintPage** (que construirá el documento a imprimir) y empezará a imprimirse en la impresora predeterminada de Windows.

Simplemente con ese código, y teniendo en el formulario un botón (button1) y el control de tipo PrintDocument (printDocument1) insertados desde el diseñador, ya imprimiría. El problema es que es un ejemplo demasiado básico, pues no muestra cómo imprimir varias páginas, cómo mostrar el cuadro de diálogo común de selección de la impresora o cómo mostrar una vista previa antes de imprimir.

FUNCIONAMIENTO DEL PROGRAMA:

1. Al cargar el formulario le asignará un nombre al documento a imprimir (en la función manejadora del evento Load de Form1). Además, indicamos al diálogo de Vista Previa que printDocument1 es el documento que debe mostrar.

```
printDocument1.DocumentName = "Mi Documento";
// Cargamos el documento a mostrar...
printPreviewDialog1.Document = printDocument1;
```

2. El documento a imprimir deberá construirse primero en **`printDocument1_PrintPage`** que es la función de evento más importante y complicada a la hora de implementar funciones de impresión en un formulario. Como ejemplo, construiremos 3 páginas a imprimir.

   Todas la páginas a imprimir se construyen dentro de esta función, y el código que construye cada página debe acabar en **`e.HasMorePages = true`** o en **`e.HasMorePages = false`**. La primera instrucción indica que ese trozo de código no es el de la última página, por tanto, la función **`printDocument1_PrintPage`** deberá ejecutarse de nuevo por sí sola (como en un bucle, y cada vez que se ejecuta crea una nueva página de impresión) hasta que encuentre la instrucción **`e.HasMorePages = false`** que indica que ese código ejecutado es el de la última página y ya no debe volver a ejecutarse más la función **`printDocument1_PrintPage`**.

   Un ejemplo simplificado (de cómo construir un documento a imprimir de varias páginas) podría ser:

```
int numeroPagina=1; // Variable Global. Inicialmente imprimiremos la Página 1.

private void printDocument1_PrintPage(object sender,
    System.Drawing.Printing.PrintPageEventArgs e)
{
    Font fuente = new Font("Arial", 12, FontStyle.Bold);

    if (numeroPagina == 1) // Creamos la primera página…
    {
        e.Graphics.DrawString("Página " + numeroPagina.ToString(),
            fuente, Brushes.Black, 150, 125); // Muestra “Página 1”
        numeroPagina++;
        e.HasMorePages = true; // Quedan páginas por construir. Bucle.
    }
    else if (numeroPagina == 2) // Creamos la segunda página…
    {
        e.Graphics.DrawString("Página " + numeroPagina.ToString(),
            fuente, Brushes.Black, 150, 125); // Muestra “Página 2”
        numeroPagina++;
        e.HasMorePages = true; // Quedan páginas por construir. Bucle.
    }
    else // Creamos la tercera y última página…
    {
        e.Graphics.DrawString("Página " + numeroPagina.ToString(),
            fuente, Brushes.Black, 150, 125); // Muestra “Página 3”
        numeroPagina = 1;
        e.HasMorePages = false; // Última página. No quedan más páginas.
    }
}
```

   En e.Graphics debe ir añadiéndose cada elemento del documento a imprimir, y siempre en unas coordenadas concretas, gracias a sus métodos: `DrawString, FillRectangle, DrawImage, etc`….

   e.HasMorePages indica, si es true, que no es la última página del documento a imprimir (por lo que se volverá a ejecutar la función **`printDocument1_PrintPage`** por sí sola), mientras que si vale false indica que esa es la última página del documento (y la función **`printDocument1_PrintPage`** no volverá a ejecutarse por sí sola más).

Controlaremos cada número de página mediante una variable global llamada numeroPagina:

```
int numeroPagina = 1;
```

3. Una vez desarrollada la función que construye el documento, para imprimirlo basta con ejecutar:

```
printDocument1.Print();
```

4. En button1_Click se muestra el cuadro de diálogo de selección de impresora, desde el que se puede incluso imprimir:

```
// Muestra el cuadro de diálogo común de Impresión.
if (printDialog1.ShowDialog() == DialogResult.OK) // Si pulsó Imprimir…
{
    Imprimir(); // Imprimir.
}
```

5. En button2_Click simplemente se imprime, llamando a la función de usuario Imprimir.
6. En button3_Click se mostrará la vista previa del documento sin necesidad de imprimirlo, aunque desde esa ventana podrá imprimirse, si se cree conveniente:

```
// Mostramos la vista previa...
printPreviewDialog1.ShowDialog();
```

7. En la función de evento **printDocument1_BeginPrint** se coloca el código que se quiera que se ejecute antes de empezar a enviar la primera página del documento a imprimir a la impresora. Y en la función **printDocument1_EndPrint** se coloca el código tras enviar una página a la impresora.

EL CÓDIGO COMPLETO DE Form1.cs:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace Imprimir
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        int numeroPagina = 1;

        // Para más información consultar esto:
        // "Funcionalidad para imprimir en formularios Windows Forms"
        // http://msdn.microsoft.com/library/spa/default.asp?url=/library/SPA/vbcon/html/vbtskprintinggraphics.asp

        private void button1_Click(object sender, EventArgs e)
        {
            // Muestra el cuadro de diálogo común de Impresión.
            if (printDialog1.ShowDialog() == DialogResult.OK)
            { // Si se pulsó el botón imprimir...
                Imprimir(); // Imprimir.
            }
        }
        private void printDocument1_PrintPage(object sender,
```

```
            System.Drawing.Printing.PrintPageEventArgs e)
        {
            // Tipo de letra, tamaño y estilo de fuente a usar.
            Font fuente = new Font("Arial", 12, FontStyle.Bold);
            // Rectángulo que queremos mostrar (left,top,ancho,alto).
            Rectangle rectangulo = new Rectangle(200, 200, 60, 20);
            // Creamos un objeto imagen y cargamos una imagen del disco:
            Image imagen = Image.FromFile("c:\\windows\\A pescar.bmp");
            Brush brocha = Brushes.Blue;

            if (numeroPagina == 1)
            {
                // PRIMERA PÁGINA (1):
                // Mostramos un texto informando del número de página.
                // DrawString(texto,fuente,brocha,left,top)
                e.Graphics.DrawString("Página " + numeroPagina.ToString(),
                    fuente, Brushes.Black, 150, 125); // Mostramos Página 1
                // Dibujamos el rectángulo relleno de azul.
                // FillRectangle(brocha,ractángulo)
                e.Graphics.FillRectangle(brocha, rectangulo);
                // Dibujamos una imagen.
                // DrawImage(imagen,left,top)
                e.Graphics.DrawImage(imagen, 250, 250);
                numeroPagina=2;
                e.HasMorePages = true; // Quedan páginas por construir.
            }
            else if (numeroPagina == 2)
            {
                // SIGUIENTE PÁGINA (2):
                // Mostramos un texto informando del número de página.
                // Insertaremos directamente un objeto Font como parámetro.
                e.Graphics.DrawString("Página " + numeroPagina.ToString(),
                    new Font("Times New Roman", 20, FontStyle.Bold)
                    , Brushes.Black, 150, 125); // Mostramos Página 2
                // Dibujamos el rectángulo relleno de rojo.
                // FillRectangle(brocha,ractángulo): en este caso creamos el
                // objeto rectánculo directamente y lo pasamos como párametro...
                e.Graphics.FillRectangle(Brushes.Red, new Rectangle(200, 200, 60, 20));
                // Dibujamos una imagen.
                // DrawImage(imagen,left,top): en este caso creamos el
                // objeto imagen directamente y lo pasamos como párametro...
                e.Graphics.DrawImage(Image.FromFile("c:\\windows\\Azteca.bmp"),
                        250, 250);
                numeroPagina++;
                e.HasMorePages = true; // Quedan páginas por construir.
            }
            else
            {
                // ÚLTIMA PÁGINA (3):
                // Mostramos un texto informando del número de página.
                e.Graphics.DrawString("Página " + numeroPagina.ToString(),
                    fuente, Brushes.Black, 150, 125); // Mostramos Página 3
                // Dibujamos el rectángulo relleno de verde.
                // FillRectangle(brocha,left,top,ancho,alto):
                e.Graphics.FillRectangle(Brushes.Red, 200, 200, 60, 20);
                // Dibujamos una imagen.
                // DrawImage(imagen,left,top)
                e.Graphics.DrawImage(imagen, 250, 250);
                numeroPagina = 1;
                e.HasMorePages = false; // Última página. No quedan más páginas.
            }
        }
        private void button2_Click(object sender, EventArgs e)
        {
```

```
            Imprimir(); // Imprimir.
        }
        // Mostrar la vista previa.
        private void button3_Click(object sender, EventArgs e)
        {
            // Mostramos la vista previa...
            printPreviewDialog1.ShowDialog();
        }
        private void Form1_Load(object sender, EventArgs e)
        {
            printDocument1.DocumentName = "Mi Documento";
            // Cargamos el documento a mostrar...
            printPreviewDialog1.Document = printDocument1;
        }
        // Cuando empieza a enviar el documento a la impresora...
        private void printDocument1_BeginPrint(object sender,
            System.Drawing.Printing.PrintEventArgs e)
        {
            toolStripStatusLabel1.Text = "Imprimiendo...";
        }
        // Cuando acaba de enviar una página a la impresora...
        private void printDocument1_EndPrint(object sender,
            System.Drawing.Printing.PrintEventArgs e)
        {
            toolStripStatusLabel1.Text = "";
            MessageBox.Show(printDocument1.DocumentName +
                " ha terminado de imprimirse.");
        }
        private void Imprimir()
        {
            printDocument1.Print(); // Imprimir.
        }
    }
}
```