

Getting Started with PrintControls for .NET

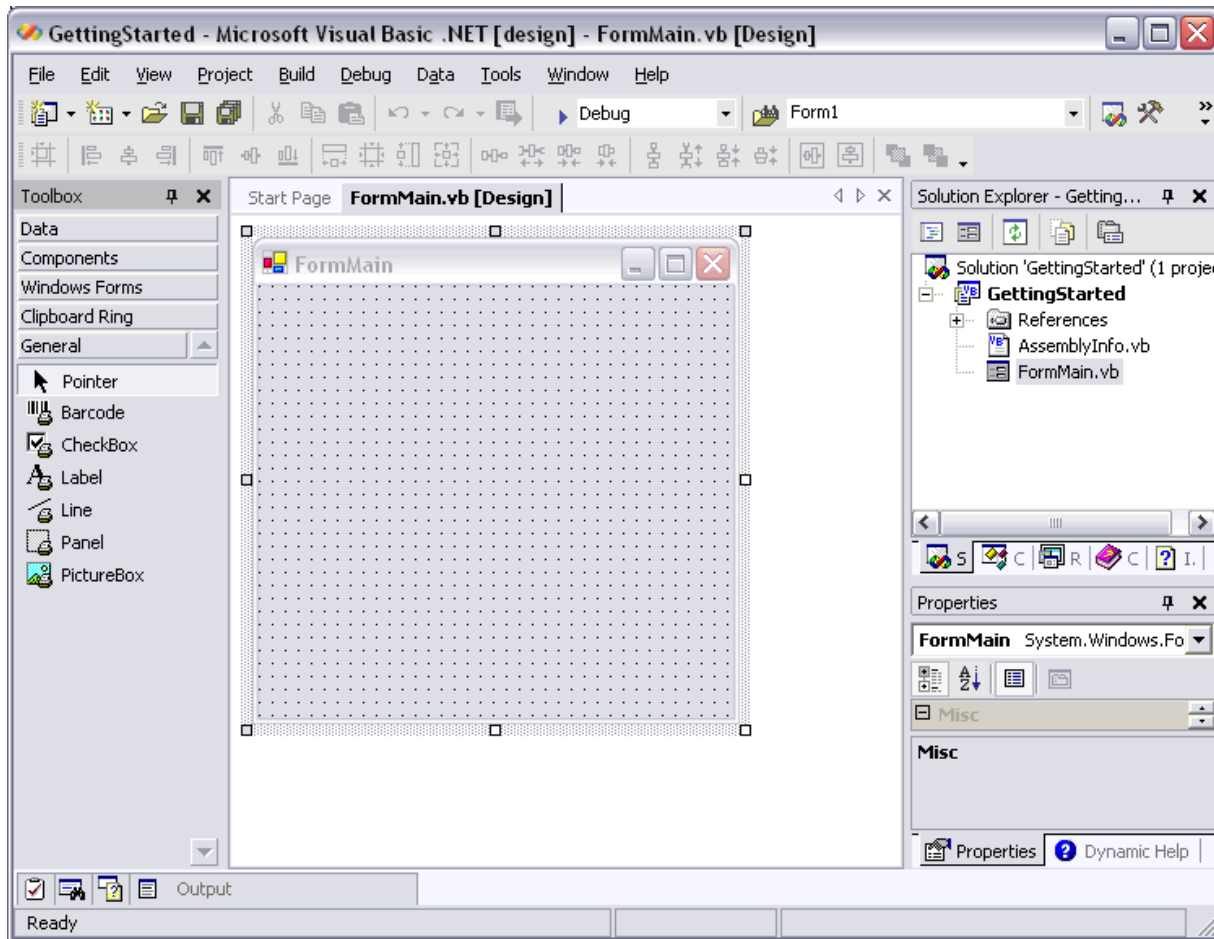
The PrintControls library provides a set of .NET Windows Forms controls that are specifically designed to produce scalable graphical output suitable for sending to the printer. This means you can design your application using our PrintControls, then when you want to print part of your form you simply call Print on the outermost container control (Panel).

Overview

This document focuses on getting started with PrintControls, aimed at the developer using PrintControls for the first time. We will create a simple form suitable for producing a mailing label, complete with barcode. The code for this sample is included with PrintControls under the Samples menu item.

Create the New Project

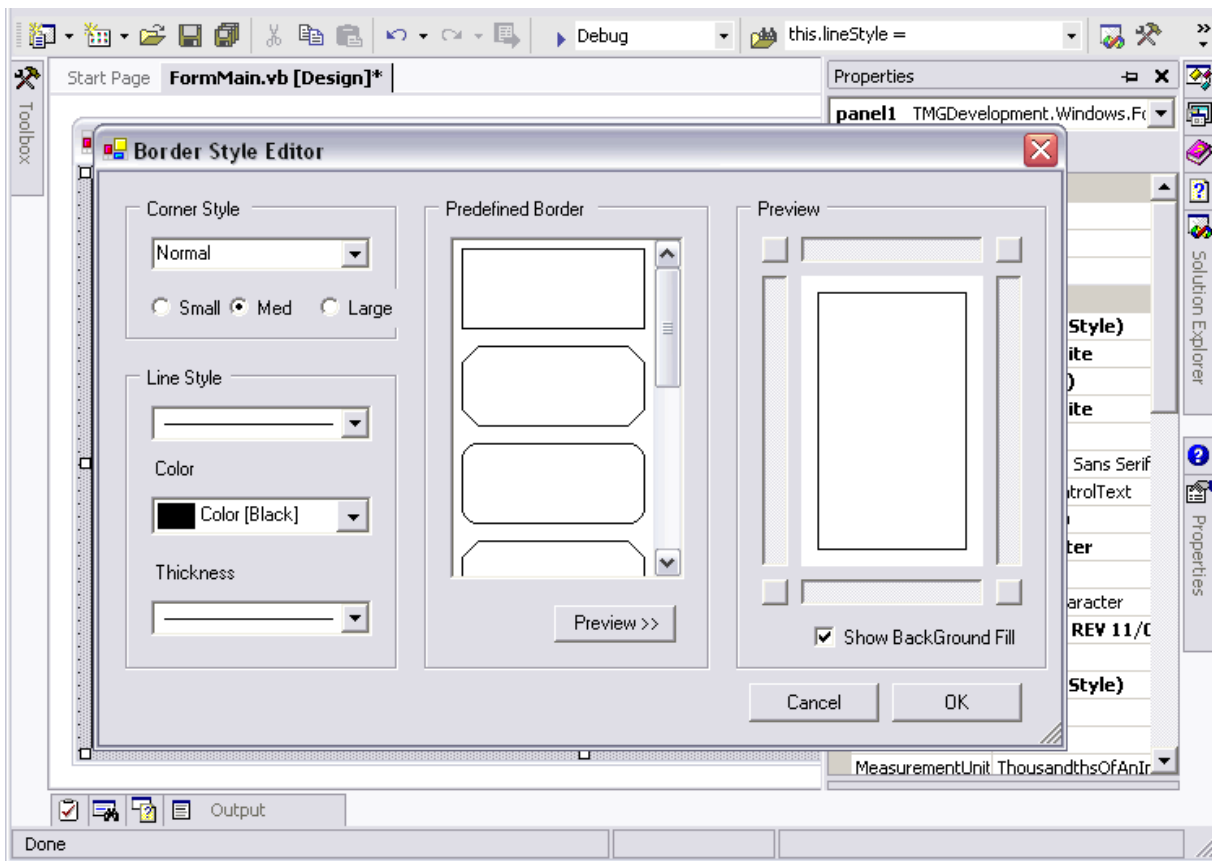
First we need a new project. Here's a new VB.NET windows forms project in the Visual Studio designer. Note we've already added the PrintControls controls to the Toolbox (see the [readme](#) for details of this):



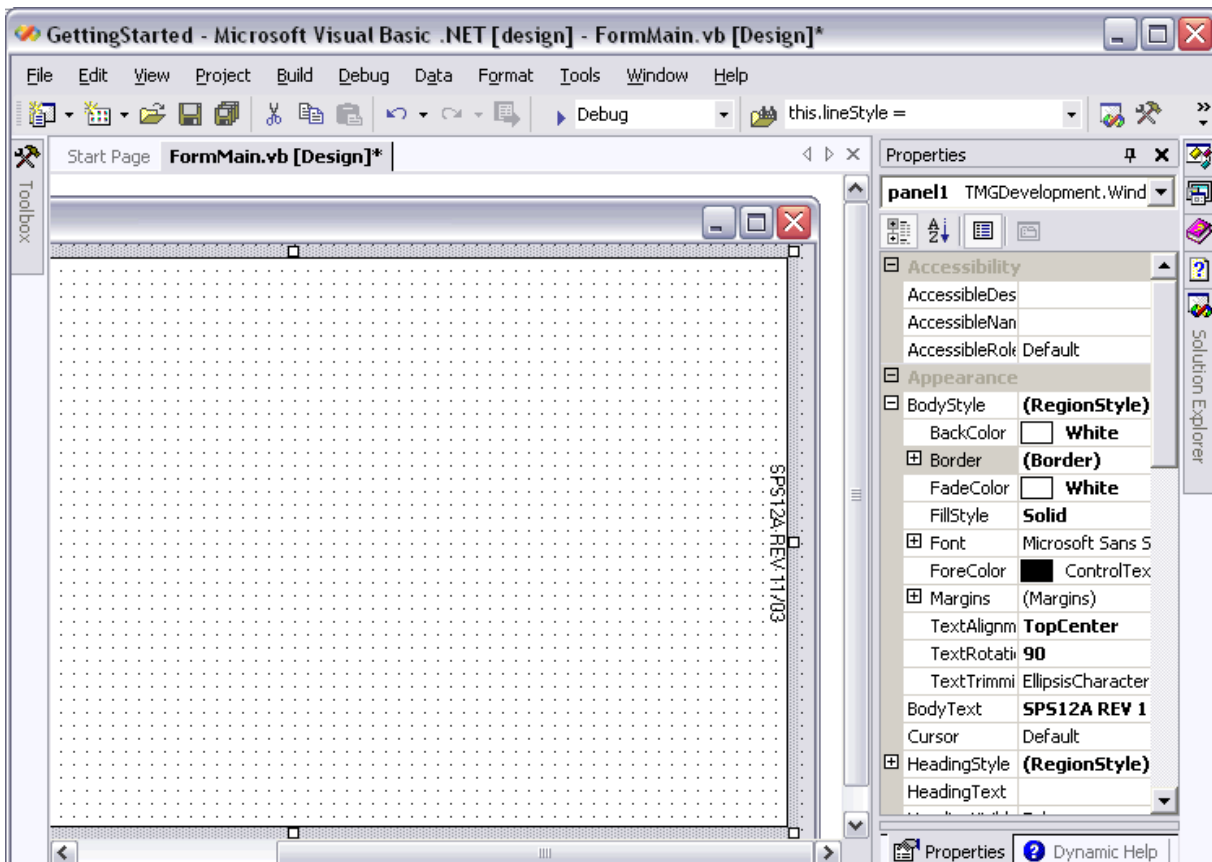
Design the Form

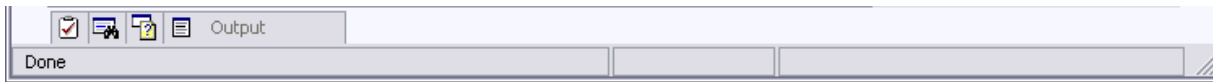
Now we'll add a PrintControl **Panel**, expand its **BodyStyle** property and click on the ellipsis next to the **Border** property, this brings up the Border Style editor. We'll set a single pixel border. Note that the Corner and Line styles selected on the left hand side of the dialog are applied by either selecting a predefined border and clicking 'Preview >>', or by clicking one of the buttons around the preview area on the right hand side. We produced the effect shown by selecting the rectangular predefined border, clicking 'Preview':



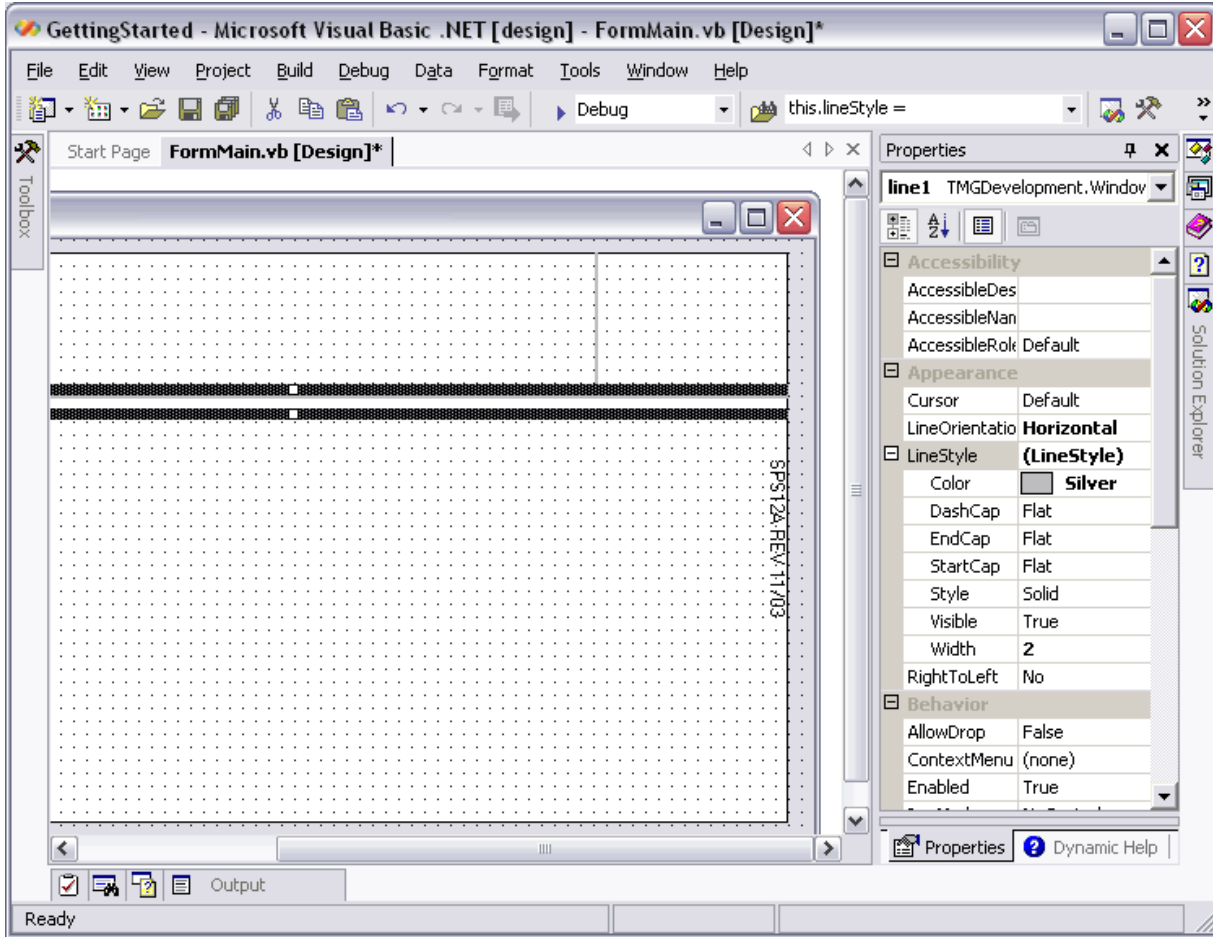


Now we enter some **BodyText** and alter the **BodyStyle TextAlignment** and **Rotation** to align the text vertically down the right hand side of the Panel.

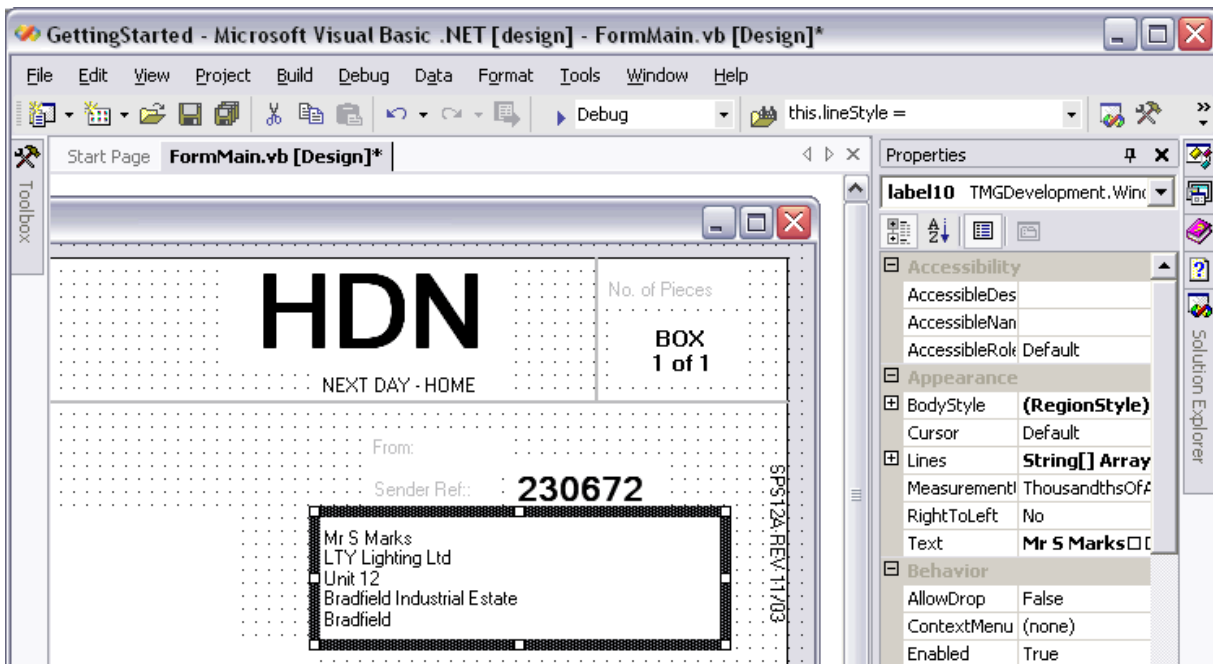


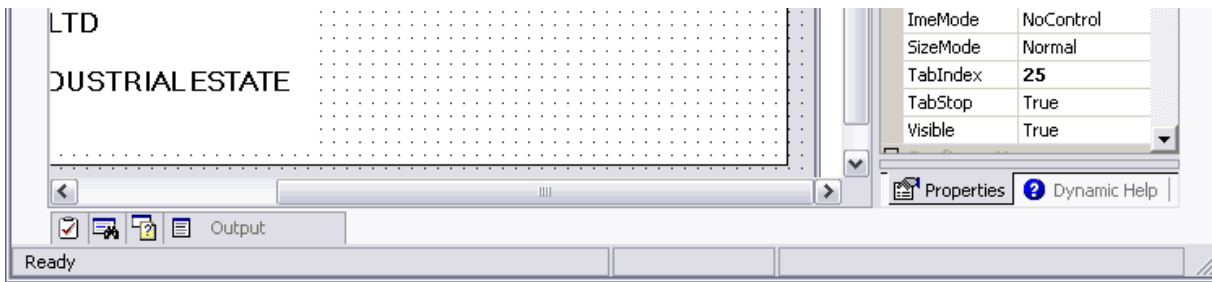


Adding a some Line controls, we set the color and thickness via the **LineStyle** property:

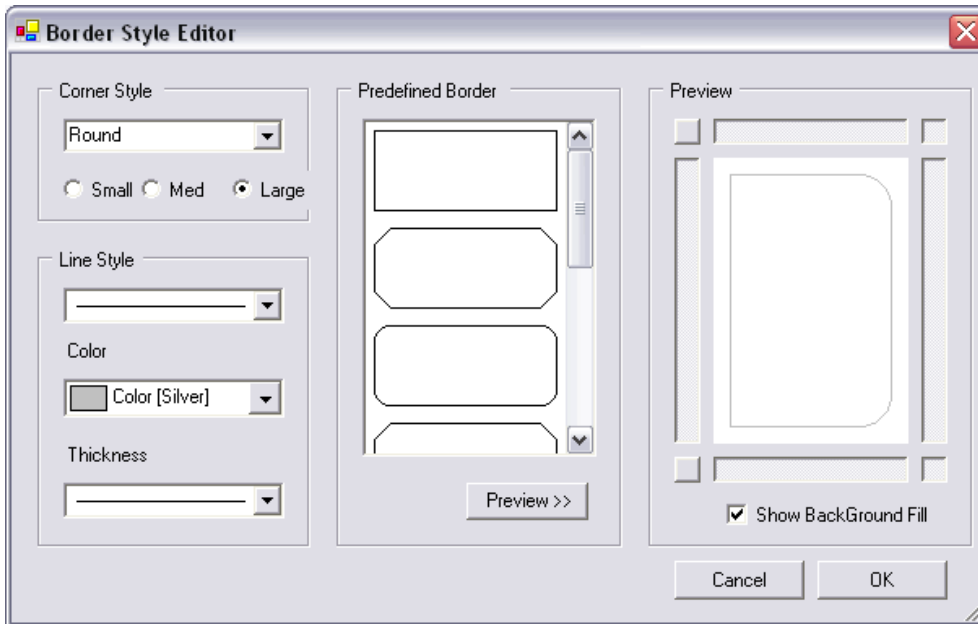


Adding a few more labels, we then highlight one to add a custom border:

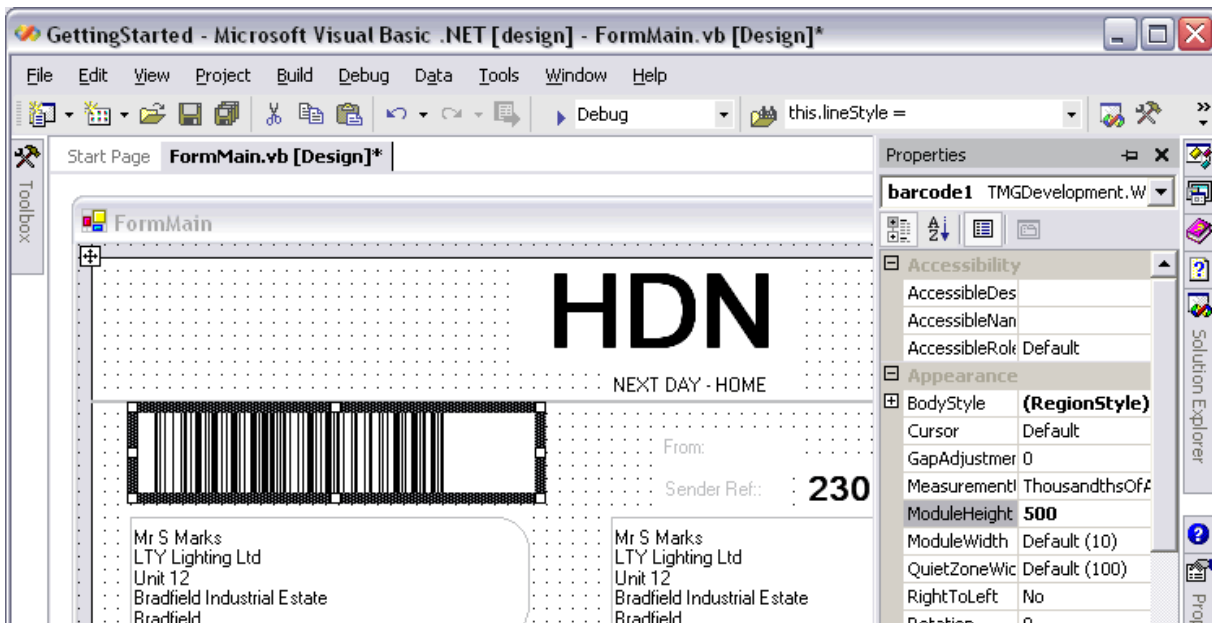




And specify the border settings via the BodyStyle Border editor as below. As before, note that the Corner and Line styles selected on the left hand side of the dialog are applied by either selecting a predefined border and clicking 'Preview >>', or by clicking one of the buttons around the preview area on the right hand side. We produced the effect shown by selecting the rectangular predefined border, clicking 'Preview', and then selecting the Round corner style and clicking the button to the top right hand and lower right corners of the preview area to apply a rounded line to that corner:

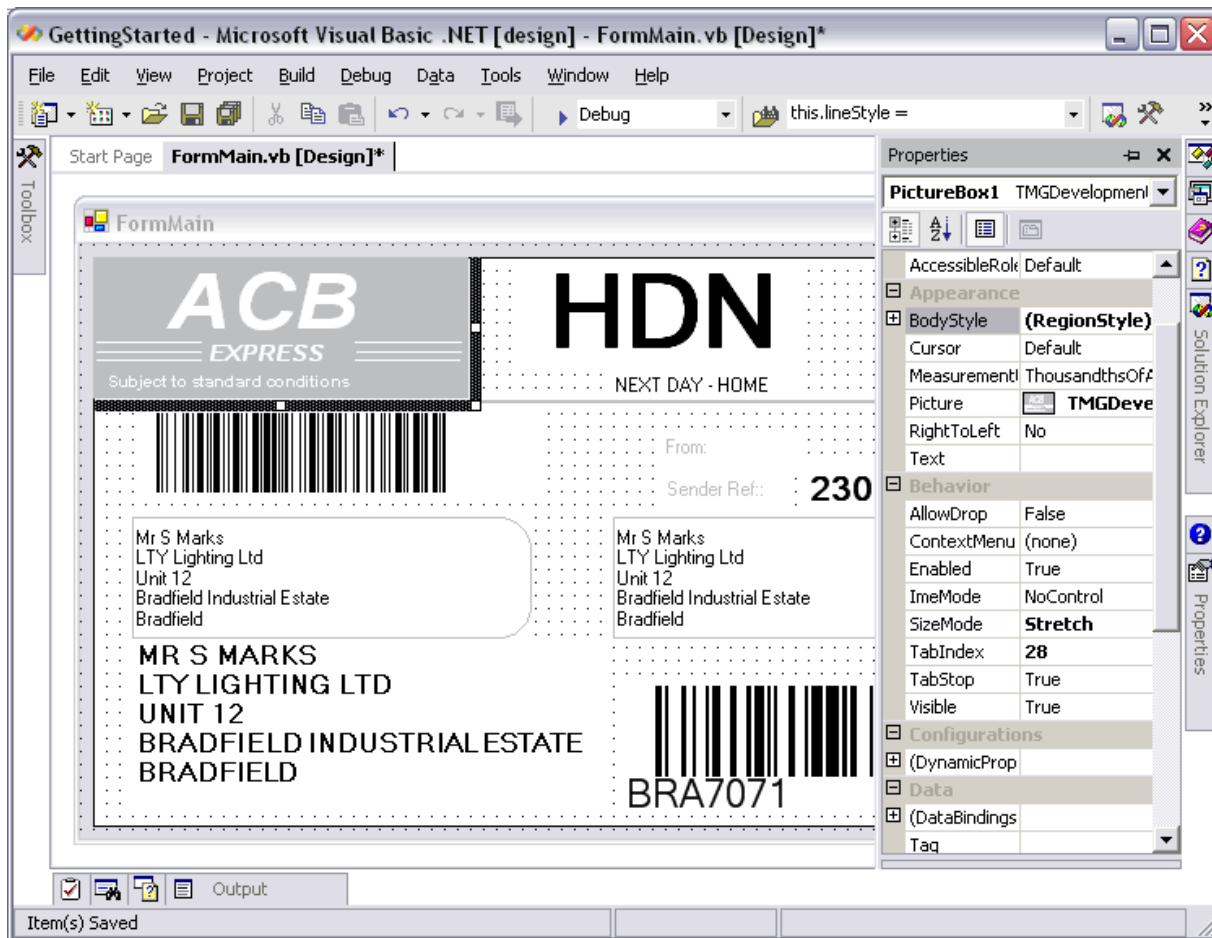


Now we add two Barcode controls to the label. Note that the Barcode provides complete control over the dimensions of the bars and gaps, expressed in actual printer units (specified by the MeasurementUnit property).





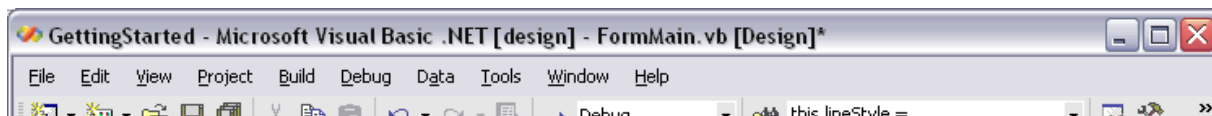
The final step is to add a logo PictureBox control to the top left of the form. Note that the PrintControls PictureBox control preserves a metafile as a metafile if you set one into its Picture property (unlike the dotnet PictureBox which translates it into a Bitmap before storing it in the resx, thereby losing the resolution independence of the metafile).



Adding the Printing Code

Printing a PrintControls Panel is a simple matter of calling its Print method, but in order to see a print preview on screen we need to add the standard .NET components PrintDocument and PrintPreviewDialog. MSDN resources will give you more details about these, but in summary all we are doing here is

1. setting the PrintPreviewDialog.Document property equal to our PrintDocument
2. adding a PrintDocument.PrintPage event handler by double clicking on the PrintDocument (adding a handler for its default event) and calling Panel.Print in it
3. adding a button with a click handler that calls PrintPreviewDialog.ShowDialog()





The code for these two event handlers looks like this:

```

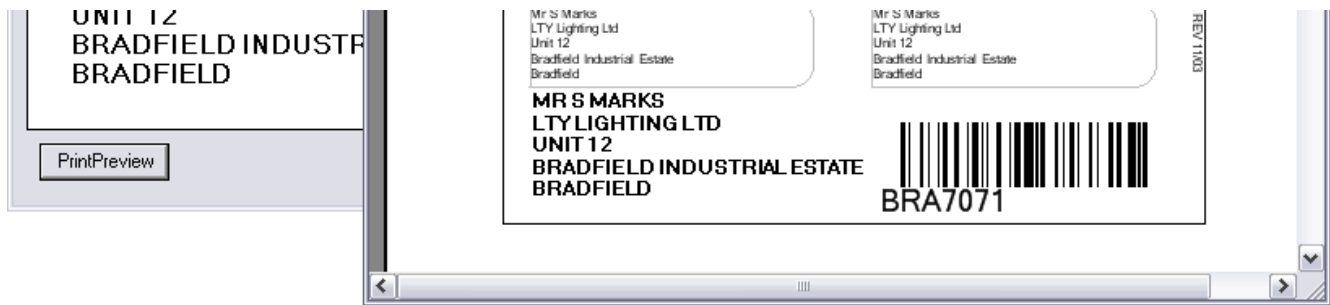
Private Sub PrintDocument1_PrintPage(ByVal sender As System.Object,
    ByVal e As System.Drawing.Printing.PrintPageEventArgs) Handles PrintDocument1.PrintPage
    Me.Panel1.Print(e.Graphics, e.MarginBounds.Location)
End Sub

Private Sub ButtonPrintPreview_Click(ByVal sender As System.Object,
    ByVal e As System.EventArgs) Handles ButtonPrintPreview.Click
    Me.PrintPreviewDialog1.ShowDialog()
End Sub
    
```

Run the Application

Hit f5 and run the app. Clicking the button gives us the screen print preview below. It's not apparent from the screenshot, but the print output is a fully scalable metafile, so whatever the resolution of your printer you'll get the best possible output. All PrintControls also support saving an image of their contents to all the usual .NET image formats (emf, gif, jpeg, bmp etc).





Summary

We've only shown a brief overview of some of the features of PrintControls here to get you started.

Other features not shown include exact printer sizing - where you set the **MeasurementUnits** of the control (1/1000th inch, 1/100th mm etc) and then set the size of the control in these units. This ensures that your barcode or panel comes out at an exact size on the printout. PrintControls also supports borders on most controls allowing specification of the corner type and size and the color, linetype, width of each section of the border. The PrintControls **PictureBox** fully supports metafile graphics (unlike the standard .NET PictureBox which converts them to a bitmap), which are especially useful on printouts as they are drawn to the printer directly giving a high resolution output. Barcodes support a variety of symbologies (Code128, Code39, ExtendedCode39, Interleaved2Of5 etc) and arbitrary rotation. Checkbox allows you to specify the check image and the box border style and fill, and there's also a simple Line control, and of course all these controls are fully printable...

For more information about PrintControls, and other .NET solutions, please visit the WinformReports website at <http://www.winformreports.co.uk>

March 10, 2005

© 2004-2005 TMG Development Ltd