Visual Studio2008 C#で
JAN13バーコードイメージを作成

Object Oriented XBASE Forum
Nobuyuki Ichioka
掲載開始日:2009年2月9日

xBASE言語をご利用の現場でバーコードの出力が必要なことが多々あります。xBASE言語製品によっては、標準でバーコード描画機能が付加されているものもあるようです。C#では、バーコードフォントを利用したり バーコードOCXや、バーコード対応レポートツールが豊富にありますので、それほど困ることは無いと思われます。

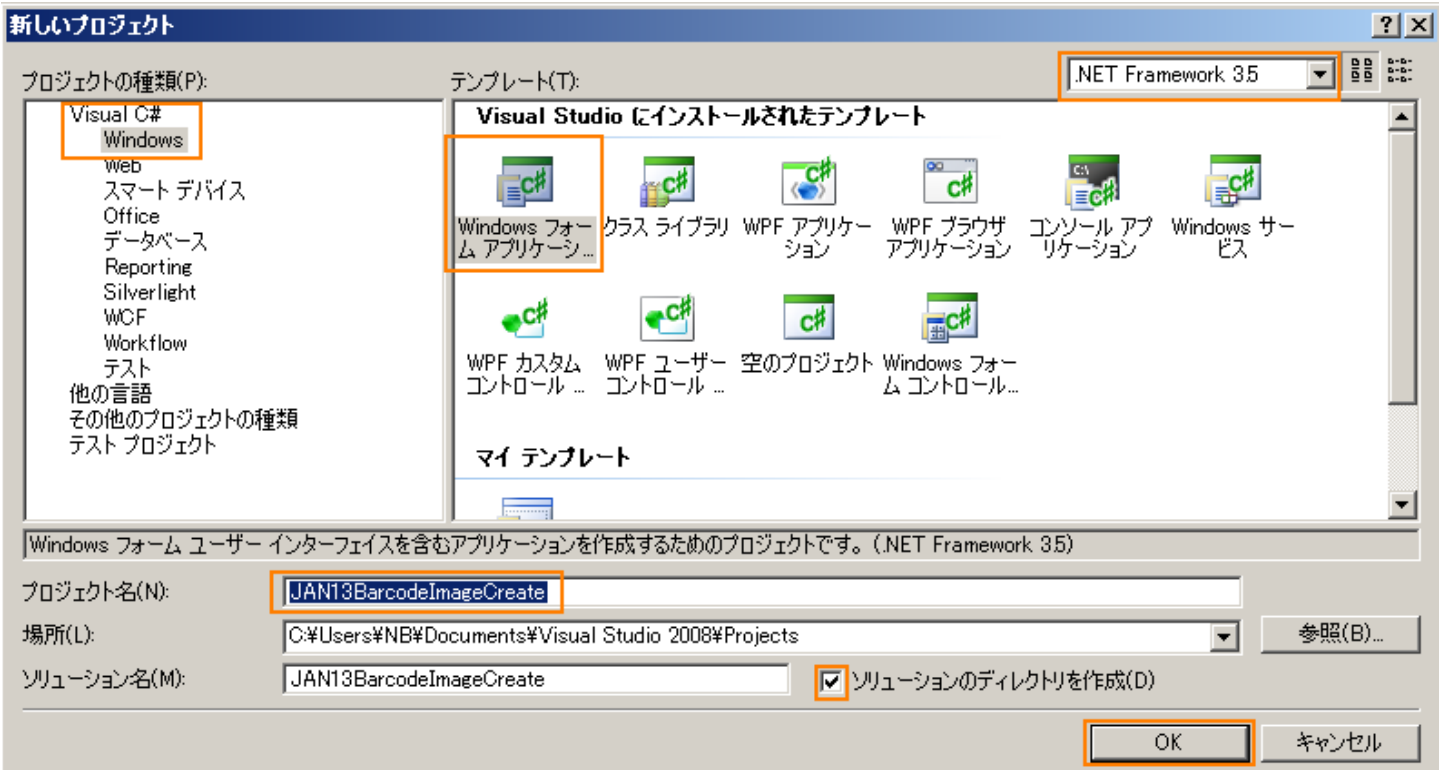
しかしJAN13程度のものでしたら、それほど難しくなくバーコードイメージを作成することが出来ます。実際にプログラミングを行ってみましょう。

ご説明には Visual Studio 2008 Professional Edition SP1を使用しました。

Visual Studioを立ち上げ新規プロジェクトを作る

それでは、VS2008を立ち上げてみましょう。

[ファイル]-[新規作成]から[プロジェクト]を選択します。【新しいプロジェクト】というダイアログが開きます。そこでプロジェクト名をJAN13BarcodeImageCreateとしてください。(名称は任意で結構です)



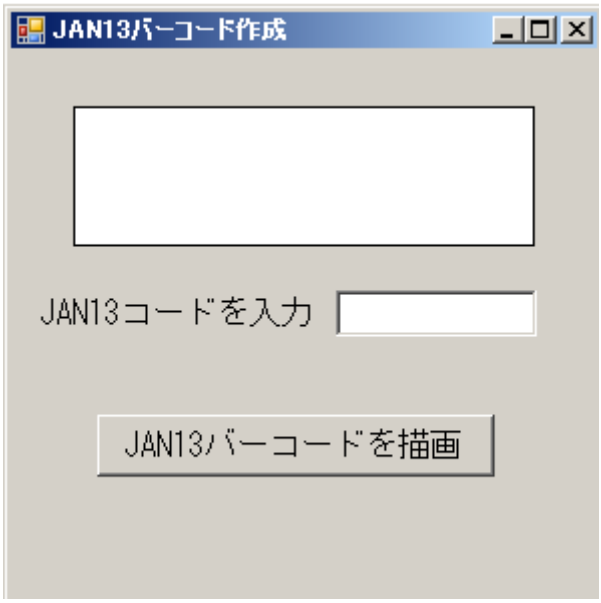
フォーム1のプロパティに”JAN13バーコード作成”と記入します。

ツールボックスから PictureBoxを選択しデザインしているフォームの上側に配置します。SizeをWidth230 Height70に、BackColorを White、BorderStyleをFixedSingleに設定します。

次にMaskedTextBoxを1つ設置し、IMEModeをDisable、MASKを9999999999999 にします。(9を13ケ) PromptCharを半角スペース1文字にします。

LABELをMaskedTextBoxのとなりに設置し プロパティのTextに“JAN13コードを入力”

と記入します。Buttonを1つ設置して、Textプロパティに”JAN13バーコードを描画”を記入してください。Button1プロパティでClickイベントをダブルクリックで、イベントのプロシージャを自動生成しておきます。



コードエディタでプログラムを書く

それでは、ひたすらコードを書いてみましょう。
コードは下記のとおりとなります。

```
Bitmap bmpBarcode;
private String[] arySPreFixPTRN = new String[10]; // プリフィックスパターンストア用
private int frmH = 0;
private int nbarX = 0;
private int nbarY = 0;
private Pen drawPen;

public Form1()
{
    InitializeComponent();

    /* この下3行を記入 */
    bmpBarcode = new Bitmap(pictureBox1.Width, pictureBox1.Height);
    pictureBox1.Image = bmpBarcode;
    InitializePreFixPattern();
}

private void InitializePreFixPattern()
{
    //************************************//
    // バーコードの左側プリフィックス対応 //
    // 奇数/偶数パリティデータ            //
    //************************************//
    arySPreFixPTRN[0] = "111111";
    arySPreFixPTRN[1] = "110100";
    arySPreFixPTRN[2] = "110010";
    arySPreFixPTRN[3] = "110001";
    arySPreFixPTRN[4] = "101100";
    arySPreFixPTRN[5] = "100110";
    arySPreFixPTRN[6] = "100011";
    arySPreFixPTRN[7] = "101010";
    arySPreFixPTRN[8] = "101001";
    arySPreFixPTRN[9] = "100101";
}
```

```
private void button1_Click(object sender, EventArgs e)
{
    if (!CheckJAN13Digit(this.maskedTextBox1.Text))
    {
        BarcodeImageClear();
        MessageBox.Show("JANコード入力エラー");
    }
    else
    {
        WriteBarcode13(this.maskedTextBox1.Text);
        string sbmpFileName = "c:¥¥～環境に合わせて～¥¥JAN13BCD"
                            + this.maskedTextBox1.Text + ".bmp";
        bmpBarcode.Save(sbmpFileName); // 自動的にBMPファイルを作成登録
    }
}

private string MakeJAN13Digit(string sBarcode12)
{
    //****************************************//
    // MakeJAN13Digit("バーコード12桁文字列")  //
    // 13桁目のチェックデジットを生成して戻す  //
    // エラーチックは実施していない            //
    //****************************************//
    int nSum = 0;
    int nI;
    char[] aryCBarcode = new char[] { };
    aryCBarcode = sBarcode12.ToCharArray();

    for ( nI = 1; nI < 12; nI += 2)
    {
        nSum = nSum+int.Parse(aryCBarcode[nI].ToString());
    }

    nSum *= 3;

    for ( nI = 0 ; nI < 12 ; nI+= 2 )
    {
        nSum = nSum+int.Parse(aryCBarcode[nI].ToString());
    }

    nSum = ((nSum/10)+1)*10 -nSum;

    return sBarcode12 + nSum.ToString();
}

private bool CheckJAN13Digit(string sBarcode13)
{
    //****************************************//
    // CheckJAN13Digit("バーコード13桁文字列") //
    // 13桁目のチェックデジットについて検証    //
    // エラー時 falseで戻す                    //
    //****************************************//
    if (sBarcode13.Length != 13)
    {
        return false;
    }
    if (sBarcode13 != MakeJAN13Digit(sBarcode13.Substring(0,12)))
    {
        return false;
    }
    return true;
}
```

```
private void BarcodeImageClear()
{
    //****************************************//
    // pictureBox1の描画データをCLEARします    //
    // グラフィックスオブジェクトを取得し      //
    // 戻り時破棄しているので他の描画プロシー  //
    // ジャ内の描画処理の間にこの関数を呼ばな  //
    // いようにします                          //
    //****************************************//
    Graphics g = Graphics.FromImage(pictureBox1.Image);
    g.Clear(Color.White);
    pictureBox1.Refresh();
    g.Dispose();
}

private void SetDrawStartPoint()
{
    //****************************************//
    // pictureBox1に描画するバーコード線の開始 //
    // 位置を設定します。                      //
    //****************************************//
    frmH = pictureBox1.Height - 26;
    nbarX = 14;
    nbarY = 4;
    drawPen = new Pen(Color.Black, 1);
}

private void WriteBarcode13(string sBarcode13)
{
    //****************************************//
    // WriteBarcode13("バーコード13桁文字列")  //
    // 13桁のバーコード数値をパラメータとして  //
    // 与える。バーコード数値についてチェック  //
    // 行っていないので事前に                  //
    // CheckJAN13Digit(string sBarcode13)を実  //
    // 行すること                              //
    // バーコードを描画するメインプロシージャ  //
    //****************************************//
    int nI;
    Font fGothic10 = new Font("ＭＳ ゴシック", 10);
    char[] aryCBarcode = new char[] { };
        aryCBarcode = sBarcode13.ToCharArray();
    int nPreFix = int.Parse(aryCBarcode[0].ToString());
    char[] PreFixPTRN = arySPreFixPTRN[nPreFix].ToCharArray();

    BarcodeImageClear();
    SetDrawStartPoint();

    Graphics g = Graphics.FromImage(pictureBox1.Image);

    g.DrawString(aryCBarcode[0].ToString(), fGothic10, Brushes.Black, nbarX - 8,
      frmH + 8);
    WriteGuardBar(g);
    for (nI = 1; nI < 7; nI++)
    {
        g.DrawString(aryCBarcode[nI].ToString(), fGothic10, Brushes.Black, nbarX ,
      frmH + 8);
        WriteEachBarLine(g, aryCBarcode[nI], PreFixPTRN[nI -1]);
    }
    WriteGuardBar(g);
    for (nI = 7; nI < 13; nI++)
    {
        g.DrawString(aryCBarcode[nI].ToString(), fGothic10, Brushes.Black, nbarX,
      frmH + 8);
        WriteEachBarLine(g, aryCBarcode[nI], '2');
    }
    WriteGuardBar(g);
    pictureBox1.Refresh();
    g.Dispose();
}
```

```
private void WriteGuardBar(Graphics g)
{
    //****************************************//
    // バーコードのガードバーを描画する。      //
    // 両端と中央に縦線を入れる                //
    // WriteBarcode13(string sBarcode13) から  //
    // 呼ばれる                                //
    //****************************************//
    nbarX = nbarX + 2;
    for (int nI = 0; nI < 2; nI++)
    {
        g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH + 16);
        nbarX = nbarX + 4;
    }

}

private void WriteEachBarLine(Graphics g, char cPos, char cPattern)
{
    //****************************************//
    // バーコード数値に応じた各バーコード線を  //
    // 描画するプロシージャへの分岐を担当      //
    // WriteBarcode13から呼ばれる              //
    //****************************************//
    switch (cPos)
    {
        case '0':
            WriteBcd0(g, cPattern);
            break;
        case '1':
            WriteBcd1(g, cPattern);
            break;
        case '2':
            WriteBcd2(g, cPattern);
            break;
        case '3':
            WriteBcd3(g, cPattern);
            break;
        case '4':
            WriteBcd4(g, cPattern);
            break;
        case '5':
            WriteBcd5(g, cPattern);
            break;
        case '6':
            WriteBcd6(g, cPattern);
            break;
        case '7':
            WriteBcd7(g, cPattern);
            break;
        case '8':
            WriteBcd8(g, cPattern);
            break;
        case '9':
            WriteBcd9(g, cPattern);
            break;
    }

}
```

```
private void WriteBcd0(Graphics g, char cPattern)
{
    //****************************************//
    // バーコード値 [0]に対応                 //
    // バーコード数値に応じた各バーコード線を  //
    // 描画するプロシージャ                   //
    // WriteEachBarLineから呼ばれる           //
    //****************************************//
    switch (cPattern)
    {
        case '0':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH );
            nbarX = nbarX + 6;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 6, frmH );
            nbarX = nbarX + 8;
            break;
        case '1':
            nbarX = nbarX + 4;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 6;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            break;
        case '2':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 6, frmH);
            nbarX = nbarX + 10;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            break;
    }
}

private void WriteBcd1(Graphics g, char cPattern)
{
    //****************************************//
    // バーコード値 [1]に対応                 //
    // バーコード数値に応じた各バーコード線を  //
    // 描画するプロシージャ                   //
    // WriteEachBarLineから呼ばれる           //
    //****************************************//
    switch (cPattern)
    {
        case '0' :
        case '2' :
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 8;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 6;
            break;
        case '1':
            nbarX = nbarX + 2;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 8;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            break;
    }
}
```

```
private void WriteBcd2(Graphics g, char cPattern)
{
    //****************************************//
    // バーコード値 [2]に対応                 //
    // バーコード数値に応じた各バーコード線を  //
    // 描画するプロシージャ                   //
    // WriteEachBarLineから呼ばれる           //
    //****************************************//
    switch (cPattern)
    {
        case '0':
            nbarX = nbarX + 2;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 6;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 6;
            break;
        case '1':
            nbarX = nbarX + 2;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 6;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 6;
            break;
        case '2':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 6;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 8;
            break;
    }
}

private void WriteBcd3(Graphics g, char cPattern)
{
    //****************************************//
    // バーコード値 [3]に対応                 //
    // バーコード数値に応じた各バーコード線を  //
    // 描画するプロシージャ                   //
    // WriteEachBarLineから呼ばれる           //
    //****************************************//
    switch (cPattern)
    {
        case '0':
        case '2':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 10;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            break;
        case '1':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 8, frmH);
            nbarX = nbarX + 10;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            break;
    }
}
```

```
private void WriteBcd4(Graphics g, char cPattern)
{
    //****************************************//
    // バーコード値 [4]に対応                 //
    // バーコード数値に応じた各バーコード線を  //
    // 描画するプロシージャ                   //
    // WriteEachBarLineから呼ばれる           //
    //****************************************//
    switch (cPattern)
    {
        case '0':
            nbarX = nbarX + 2;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 6, frmH);
            nbarX = nbarX + 8;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            break;
        case '1':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 8;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 6;
            break;
        case '2':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 6, frmH);
            nbarX = nbarX + 10;
            break;
    }
}

private void WriteBcd5(Graphics g, char cPattern)
{
    //****************************************//
    // バーコード値 [5]に対応                 //
    // バーコード数値に応じた各バーコード線を  //
    // 描画するプロシージャ                   //
    // WriteEachBarLineから呼ばれる           //
    //****************************************//
    switch (cPattern)
    {
        case '0':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 6, frmH);
            nbarX = nbarX + 10;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            break;
        case '1':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 10;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            break;
        case '2':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 6;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 6, frmH);
            nbarX = nbarX + 8;
            break;
    }
}
```

```
private void WriteBcd6(Graphics g, char cPattern)
{
    //****************************************//
    // バーコード値 [6]に対応                 //
    // バーコード数値に応じた各バーコード線を  //
    // 描画するプロシージャ                   //
    // WriteEachBarLineから呼ばれる           //
    //****************************************//
    switch (cPattern)
    {
        case '0':
            nbarX = nbarX + 6;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            break;
        case '1':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 8, frmH);
            nbarX = nbarX + 10;
            break;
        case '2':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 10;
            break;
    }
}

private void WriteBcd7(Graphics g, char cPattern)
{
    //****************************************//
    // バーコード値 [7]に対応                 //
    // バーコード数値に応じた各バーコード線を  //
    // 描画するプロシージャ                   //
    // WriteEachBarLineから呼ばれる           //
    //****************************************//
    switch (cPattern)
    {
        case '0':
            nbarX = nbarX + 2;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 8;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            break;
        case '1':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 6, frmH);
            nbarX = nbarX + 8;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 6;
            break;
        case '2':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 8;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 6;
            break;
    }
}
```

```
private void WriteBcd8(Graphics g, char cPattern)
{
    //****************************************//
    // バーコード値 [8]に対応                 //
    // バーコード数値に応じた各バーコード線を  //
    // 描画するプロシージャ                   //
    // WriteEachBarLineから呼ばれる           //
    //****************************************//
    switch (cPattern)
    {
        case '0':
            nbarX = nbarX + 4;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 6;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            break;
        case '1':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 6;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 6, frmH);
            nbarX = nbarX + 8;
            break;
        case '2':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 6;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 8;
            break;
    }
}

private void WriteBcd9(Graphics g, char cPattern)
{
    //****************************************//
    // バーコード値 [9]に対応                 //
    // バーコード数値に応じた各バーコード線を  //
    // 描画するプロシージャ                   //
    // WriteEachBarLineから呼ばれる           //
    //****************************************//
    switch (cPattern)
    {
        case '0':
            nbarX = nbarX + 2;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 6, frmH);
            nbarX = nbarX + 8;
            break;
        case '1':
            nbarX = nbarX + 4;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 4;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 4, frmH);
            nbarX = nbarX + 6;
            break;
        case '2':
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 6, frmH);
            nbarX = nbarX + 8;
            g.FillRectangle(Brushes.Black, nbarX, nbarY, 2, frmH);
            nbarX = nbarX + 6;
            break;
    }
}

// -------------------- END of Procedure -------------------------//
```

プログラムを実行してみましょう

コードを書着終わりましたら、テストRUNを行ってみます。
適当な商品についている JAN13コードを入力してボタンを押してみてください。

上図のようにバーコードが描画されましたでしょうか。商品のバーコードと比べてみて同じように描かれていれば完成です。
お疲れ様でした。