WinForm版本ComponentOneBarCode概述

通过WinForm版本的ComponentOne BarCode，您可以添加条码图像至单元格，网页，或者常规的.NET PrintDocument对象。
和普通的条码字体不同，WinForm版本的BarCode会基于不同的条码类型，根据进行编码的值，添加必要的控制字符和校验字符，以减少读取错误。您同样也可以将WinForm版本的BarCode像常规的程序集一样部署在您的应用程序中。因为它是一个免版税的DLL，您不必像安装条码字体那样需要在将其安装到客户端机器时需要确认他们是否是免费的。
WinForm版本的BarCode非常易于使用，您只需要添加该控件到您的窗体，设置条码的编码类型，之后就可以立即投入使用了！

入门

开始之前，回顾以下主题：

- 主要功能
- 快速入门
- 示例

ComponentOne Studio WinForms帮助

入门

关于安装ComponentOne Studio for WinForms，许可授权，技术支持，命名空间以及使用控件创建一个工程的更多信息，请访问Getting Started with Studio for WinForms.章节。

新功能

关于添加到ComponentOne Studio for WinForms最新功能的列表，请访问What's New in Studio for WinForms。

迁移一个WinForm版本的ComponentOneBarCode工程到Visual Studio

为了将一个使用ComponentOne的工程进行迁移首先，必须把您的工程转换为Visual Studio格式，这包括移除对之前版本程序集的引用以及替换新的程序集。其次，必须更新.licx文件或者许可授权文件，以便工程能够正确运行。

转换工程的步骤:

1. 打开Visual Studio，选择File菜单，Open Project菜单项。
2. 找到您希望转换为Visual Studio的.sln文件。选中并单击Open。之后将会出现Visual Studio转换向导。
3. 单击Next。
4. 选择Yes，在转换之前创建一个您当前工程的备份，并单击Next。
5. 单击Finish按钮，将您的工程转换为Visual Studio格式。转换完成窗口出现。

6. 如果您希望查看转换记录，请单击选择“向导关闭时显示转换记录”。
7. 单击Close。工程转换完成并打开。现在你必须删除对任何以前版本的ComponentOne .dll文件的引用，并添加对新版本的引用。
8. 转到“解决方案资源管理器”（View |Solution Explorer），选择该工程，然后单击ShowAllFiles按钮。

> **注意：**如果没有选中Solution中的工程节点，则不会在SolutionExplorer工具栏显示Show All Files按钮。

9. 展开References节点，右键单击C1.Common并选择移除。使用同样的方法移除C1.Win.C1BarCode.2.dll。
10. 右键单击Reference节点，并选择AddReference。
11. 浏览并选择C1.Win.C1BarCode.2.dll。单击OK，将其添加到该工程的引用。

更新.licx文件：

1. 在解决方案资源管理器中，右键单击licenses.licx文件并选择删除。
2. 单击OK永久删除licenses.licx必须重新Build整个工程以便创建一个新的，内容更新过后的.licx文件。
3. 单击Start Debugging按钮开始编译并运行工程。新增的.licx文件可能在解决方案资源管理器中不可见。
4. 选择File，Close以关闭窗体，然后双击Form.vb或Form.cs文件
   在解决方案资源管理器中打开它。新的licenses.licx文件将会出现在文件列表中。

迁移过程到此结束。

主要功能

C1BarCode 从最早就是为了在C1Report组件中使用而开发的，并且自从2004年五月发布的build 151开始做为C1Report的一部分包含进来。

我们认为，ComponentOne BarCode for WinForms可以做为ComponentOne的一个有价值的补充我们决定它作为一个独立的组件，因为除了报表之外，对于其他的应用程序而言，它也很有用。例如，您可以使用C1BarCode向Grid单元格、网页、或者其他常见的.NET PrintDocument添加条码图像。下面的图片显示了一个C1Report 报表，显示的每一条数据都包含C1BarCode：

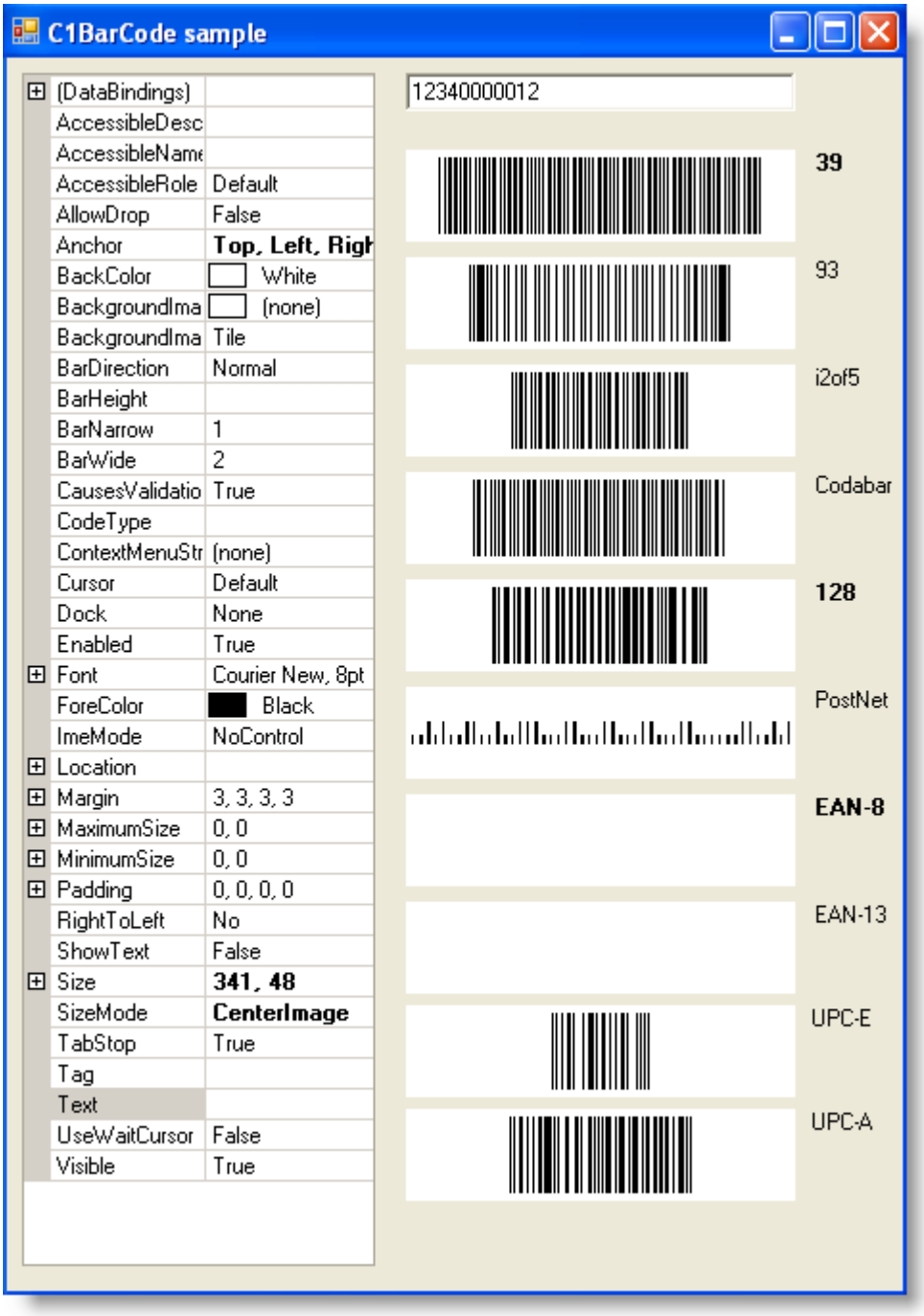
Products Report

ProductID	*ProductName*	*UnitPrice*	*UnitsInSt*	*C1BarCode*
1	Chai	18	39	
2	Chang	19	17	
3	Aniseed Syrup	10	13	
4	Chef Anton's Cajun Seasoning	22	53	
5	Chef Anton's Gumbo Mix	21.35	0	
6	Grandma's Boysenberry Spread	25	120	
7	Uncle Bob's Organic Dried Pears	30	15	

WinForms条码的主要功能包括：

- 支持10种不同的编码

控件支持10种编码，包括：CODABAR码，cod128，CODE39，code93，EAN13，EAN8，Code I2of5，Postnet，UPCA、UPCE。



- 提供C1QRCode 格式

  QR码（快速响应矩阵码）格式是时下最流行的二维条码格式，几乎所有的智能手机都带有免费的扫描程序，可以扫描并识别该条码。更多信息请参阅使用C1QRCode 章节。

- 自动添加校验位

  C1BarCode控件会按照不同编码的编码规则，依据需要进行编码的值，自动添加所需要的控制字符也校验字符，以保证条码能够更良好的被设备识别。

- 易于部署免版税的DLL

  C1BarCode是一个免版税的DLL，可以向任何常规程序集一样和您的应用程序一起部署。

BarCode for WinForms快速入门

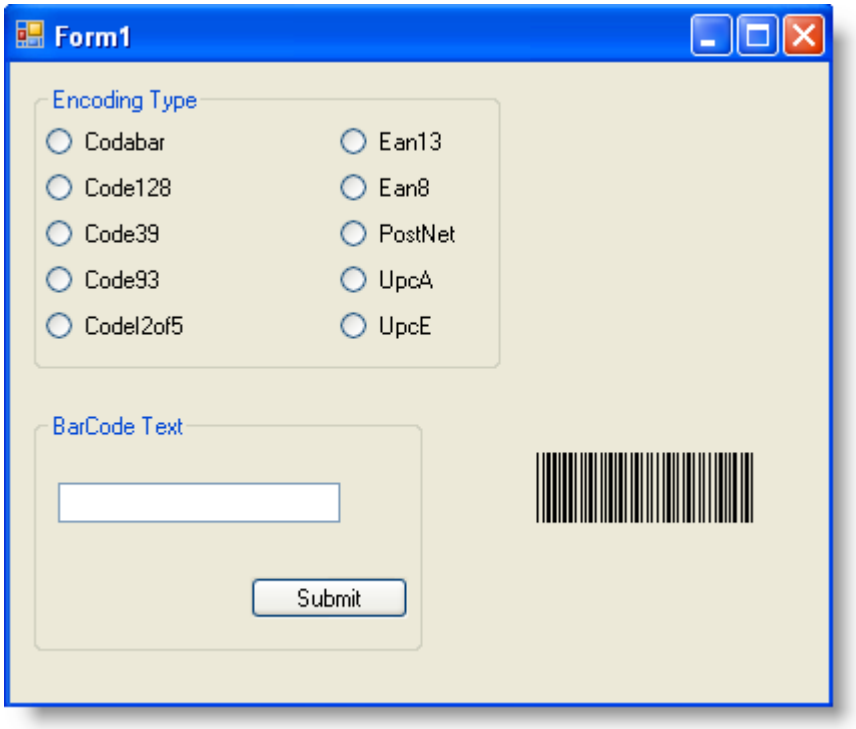
本节将详细介绍一些ComponentOne BarCode for WinForms的功能本快速入门将通过快速带您浏览如何向您的工程添加C1BarCode以及设置C1BarCode的外观和行为设置。在快速入门中创建的工程将向您演示ComponentOne BarCode for WinForms中各种可用的编码。有关可用的编码的更多信息，参见支持的编码。

步骤一：设置窗体

首先，我们将向窗体添加C1BarCode并设置该工程。完成以下步骤:

1. 创建一个新的.NET工程。
2. 从工具栏，在窗体上添加下列控件：
   - 两个GroupBox控件
   - 十个RadioButton控件
   - 一个文本框控件
   - 一个命令按钮
   - 一个C1BarCode 控件
3. 设置以下控件的文本属性，并按照如下图像排布这些控件：

名称	文本属性
GroupBox1	Encoding Type
GroupBox2	Bar Code Text
Button1	Submit
RadioButton1	CodeBar
RadioButton2	Code128
RadioButton3	Code39
RadioButton4	Code93
RadioButton5	CodeI2of5
RadioButton6	Ean13
RadioButton7	Ean8
RadioButton8	PostNet
RadioButton9	UpcA
RadioButton10	UpcE



4. 在属性窗体中，设置C1BarCode1的BackColor属性的值为Transparent。
5. 调整C1BarCode1控件的大小。

你刚刚完成使用BarCode for WinForms的第一步。继续下一步，向工程添加代码。

步骤二：将代码添加到项目

在上一步中您已经设置了窗体并向工程添加了一个C1BarCode，您将继续向工程添加代码。
完成以下步骤：

- 双击“Button1”切换到代码视图，添加Button1_Click事件。将下面的代码添加到Button1_Click事件处理，以设置C1BarCode1的文本为TextBox1中间输入的文本。

  Visual Basic

  ```
  C1BarCode1.Text = TextBox1.Text
  ```

  C#

  ```
  c1BarCode1.Text = TextBox1.Text;
  ```

- 添加下面的代码来处理单选按钮选择行为：

  注意：您必须向Form代码的开始添加 Imports C1.Win.C1BarCode（Visual Basic 工程）或者using C1.Win.C1BarCode（C#工程）以便使得以下代码正常工作。

  Visual Basic

  ```
  Private Sub RadioButton1_CheckedChanged(ByVal sender As System.Object, ByVal e
  ```

```
As System.EventArgs) Handles RadioButton1.CheckedChanged
    C1BarCode1.CodeType = CodeTypeEnum.Codabar
End Sub

Private Sub RadioButton2_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles RadioButton2.CheckedChanged
    C1BarCode1.CodeType = CodeTypeEnum.Code128
End Sub

Private Sub RadioButton3_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles RadioButton3.CheckedChanged
    C1BarCode1.CodeType = CodeTypeEnum.Code39
End Sub

Private Sub RadioButton4_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles RadioButton4.CheckedChanged
    C1BarCode1.CodeType = CodeTypeEnum.Code93
End Sub

Private Sub RadioButton5_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles RadioButton5.CheckedChanged
    C1BarCode1.CodeType = CodeTypeEnum.CodeI2of5
End Sub

Private Sub RadioButton6_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles RadioButton6.CheckedChanged
    C1BarCode1.CodeType = CodeTypeEnum.Ean13
End Sub

Private Sub RadioButton7_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles RadioButton7.CheckedChanged
    C1BarCode1.CodeType = CodeTypeEnum.Ean8
End Sub

Private Sub RadioButton8_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles RadioButton8.CheckedChanged
    C1BarCode1.CodeType = CodeTypeEnum.PostNet
End Sub

Private Sub RadioButton9_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles RadioButton9.CheckedChanged
    C1BarCode1.CodeType = CodeTypeEnum.UpcA
End Sub

Private Sub RadioButton10_CheckedChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles RadioButton10.CheckedChanged
    C1BarCode1.CodeType = CodeTypeEnum.UpcE
End Sub
```

C#

C#

```
private void radioButton1_CheckedChanged(object sender, System.EventArgs e)
{
   c1BarCode1.CodeType = CodeTypeEnum.Codabar;
}

private void radioButton2_CheckedChanged(object sender, System.EventArgs e)
{
   c1BarCode1.CodeType = CodeTypeEnum.Code128;
}

private void radioButton3_CheckedChanged(object sender, System.EventArgs e)
{
   c1BarCode1.CodeType = CodeTypeEnum.Code39;
}

private void radioButton4_CheckedChanged(object sender, System.EventArgs e)
{
   c1BarCode1.CodeType = CodeTypeEnum.Code93;
}

private void radioButton5_CheckedChanged(object sender, System.EventArgs e)
{
   c1BarCode1.CodeType = CodeTypeEnum.CodeI2of5;
}

private void radioButton6_CheckedChanged(object sender, System.EventArgs e)
{
   c1BarCode1.CodeType = CodeTypeEnum.Ean13;
}

private void radioButton7_CheckedChanged(object sender, System.EventArgs e)
{
   c1BarCode1.CodeType = CodeTypeEnum.Ean8;
}

private void radioButton8_CheckedChanged(object sender, System.EventArgs e)
{
   c1BarCode1.CodeType = CodeTypeEnum.PostNet;
}

private void radioButton9_CheckedChanged(object sender, System.EventArgs e)
{
   c1BarCode1.CodeType = CodeTypeEnum.UpcA;
}

private void radioButton10_CheckedChanged(object sender, System.EventArgs e)
{
   c1BarCode1.CodeType = CodeTypeEnum.UpcE;
}
```

你刚刚完成ComponentOne BarCode for WinForms快速入门的第二步。继续下一步，运行并查看工程运行结果。

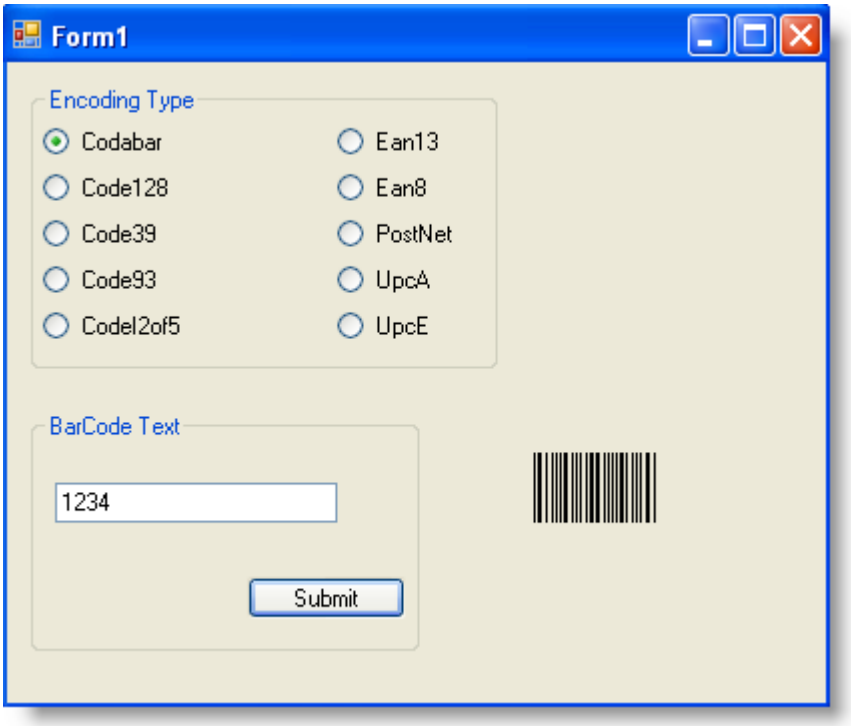
步骤三：运行项目

现在我们已经设置好了工程并添加了代码，我们将运行该工程并查看由ComponentOne BarCode for WinForms支持的不同编码。

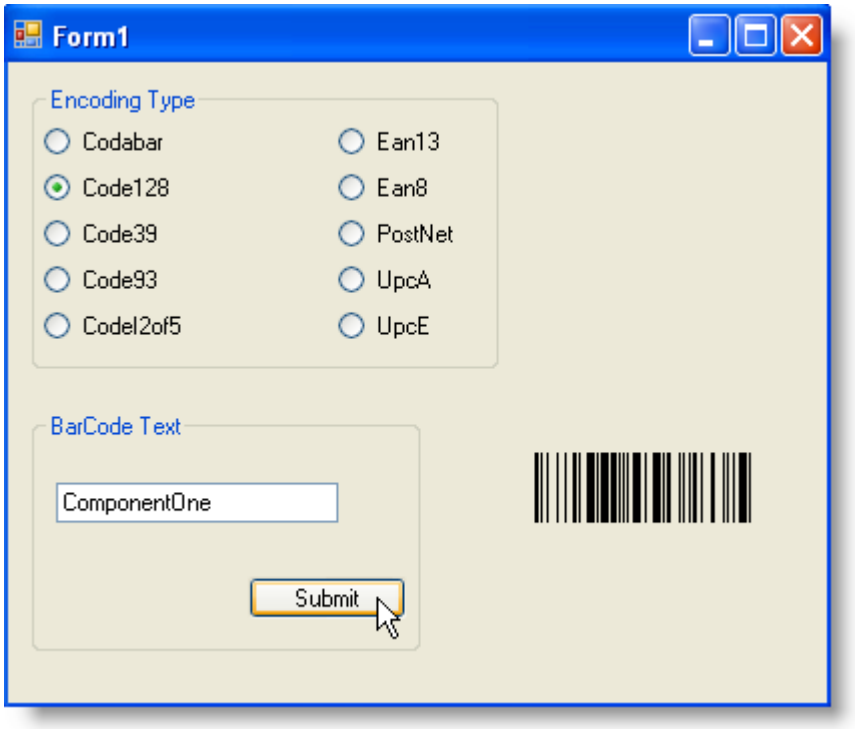
完成以下步骤：

1. 运行该项目并选择CodeBar单选按钮。
2. 输入一个数字的字符串，例如1234，在文本框中，点击提交按钮。

   条码现在看起来类似于下面这样：



3. 选择不同的单选按钮改变编码类型，您可以注意到C1BarCode条形码的外观将发生变化。
4. 尝试输入您的名字或者其他不同的字母和数字字符组成的字符串，以查看不同的编码类型能够接受哪些类型的字符：



请注意，某些编码有一个字符最小长度的需求，同时有些编码只对数字类型的值有效。关于编码的更多信息，参见支持的编码。

到此我们结束ComponentOne BarCode for WinForms快速入门章节。

使用BarCode for WinForms

以下章节将进一步详细解释如何使用ComponentOne BarCode for WinForms，包括支持的编码格式的详细信息。

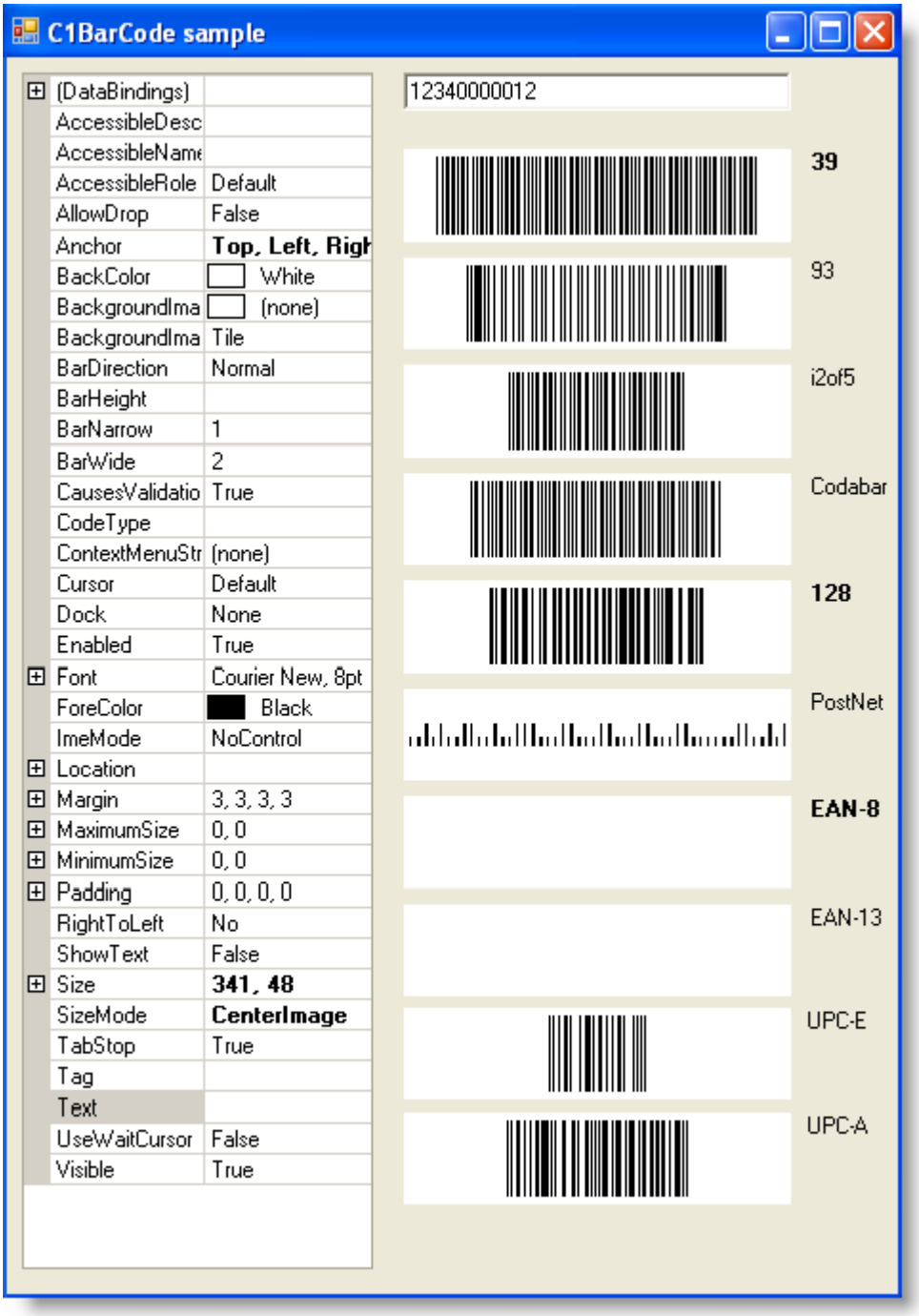
支持的编码

您可以通过设置CodeType属性改变C1BarCode编码类型。C1BarCode 控件支持以下编码：

编码	描述
Code Bar	CodeBar可以输入16种不同的字符（0到9，外加 -$:/.+），并支持额外的4种不同的开始/结束字符（A到D）。CodeBar被美国的一些血库、照片冲印店以及联邦快递airbill采用。
Code128	Code128是一中非常高密度的字母数字条码。在全部现有的一维条码中它占用的空间最小，支持6位或者更多的字符。
Code39	Code39是一种支持字母和数字的编码，也称作三九条码以及LOGMARS。这是开发的第一种支持字母数字的条码，是目前应用最广泛的条形码之一。
Code93	Code93也是一种支持字母和数字的编码，其密度要稍高于Code39。
CodeI2of5	Code I2of5是一个仅支持数字的编码。是一种可以按照需要存储的编码持续增长的条形码。
Ean13	EAN-13条码是由国际物品编码协会（EAN）在欧洲实施的条形码。EAN-13对一个十二位的数字进行编码，该数字串包括两位的系统代码，紧接着是五位数字的表示制造商的编码以及五位的产品编码。十二位数字位代码末尾将跟着一个校验位（由控件自动添加）。
Ean8	EAN-8 为小型包装提供一个短小的条形码。它将对七位数字进行编码，代码包括两位或三位的系统编码跟随着四位或五位的产品编码。七位数字末尾跟着一个校验位（由控件自动添加）。
PostNet	PostNet是一个由美国邮政局使用的数字编码。和大多数的其他条码不同，它是基于条形的高度表示信息而大多数条形码是靠靠宽度表示信息。
UpcA	UPC-A是一种常见的编码，您会在超市货架上的几乎每一种消费品上见到，其他的还包括书籍，杂志，以及报纸。它类似于EAN-13，由11位数字和末尾的校验位组成。
UpcE	UPC-E是UPC-A条码的一种变体，通过消除“多余”的零，生成更紧凑的编码。由于UPC-E条码是大约半个UPC-A条码的尺寸，它一般用于非常小的包装产品。使用UPC-E编码时，设置文本属性为一个11位的字符串，就像是在使用UPC-A编码一样。注意：不是所有的UpcA代码可以被UpcE进行编码。如果制造商代码以“000”，“100”，或“200”结尾，则产品码的数值必须≤900。如果制造商代码以“00”结尾而不是“100”，“200”，或“300”，那么产品码的数值必须≤90。如果制造商代码以“0”结尾而不是“00”，那么产品码的数值必须≤9。如果制造商代码并没有以“0”结尾，那么产品数码的数值必须在5和9之间。

自定义C1BarCode 控件

使用C1BarCode 控件，首先要通过CodeType属性设置希望使用的编码类型，之后设置Text 属性为希望进行编码的值。



控件将显示条形码图像。注意有些编码有最短字符长度要求，而其他的一些仅支持数字字符表示的字符串值。

在文档中使用C1BarCode控件

如果您希望在一个文档中包含该条形码，请使用Image 属性获取一个条形码的图像分辨率无关的图像。例如：

Visual Basic

Visual Basic

```
C1BarCode1.CodeType = C1.Win.C1BarCode.CodeTypeEnum.Code39
C1BarCode1.Text = "123456"
PictureBox1.Image = C1BarCode1.Image
```

C#

C#

```
c1BarCode1.CodeType = CodeTypeEnum.Code39;
c1BarCode1.Text = "123456";
pictureBox1.Image = c1BarCode1.Image;
```

使用C1QRCode

QR码（快速响应矩阵码）格式是时下最流行的二维条码格式之一，几乎所有的智能手机都带有免费的扫描程序，可以扫描并识别该条码。由DENSO-WAVE公司开发的QR码的格式是高效且紧凑的，它不一定需要一个特殊的扫描枪读取它。同时它是一个开放的、免费的标准（ISO / EC18004及其他）。

该条码简单地由黑色和白色的像素点组成。通过使用C1QRCode 控件，QR条码按照您希望进行编码的值生成，该值可以通过Text属性进行设置。

如何从C1QRCode获得图像

C1QRCode控件用来显示QR图像。如果您希望在文档中包含一个QR二维码的图像，您可以通过C1QRCode.Image属性获取该条形码的图像。

请按照以下步骤使用C1QRCode控件：

1. 从Visual Studio的工具栏拖拽一个C1QRCode控件至设计窗体
2. 在属性窗体中，设置C1QRCode的C1QRCode.Text属性为希望进行编码的值。QR条码将基于C1QRCode.Text生成的值产生条码。

请参看随产品提供的QRCodeConsructor示例。示例的默认安装位置位于StudioforWinForms\C1BarCode\VB以及CS文件夹，附属于C:\Users\<username>\Documents\ComponentOneSamples(Windows7/Vista)或者C:\DocumentsandSettings\<username>\MyDocuments\ComponentOneSamples(WindowsXP)。

本示例演示如何将输入的格式化信息转换为可以打印或保存的QR二维码。

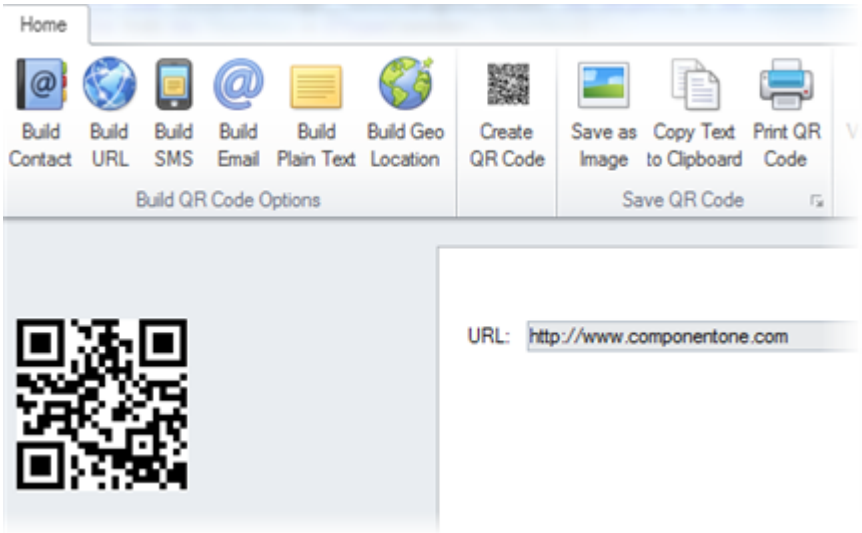
例如，以下代码整理输入的URL文本，返回一个删除了前导和末尾空白字符的字符串；同时检查文本确保长度小于271个字符；最后基于输入的文本创建QR二维码图像。

Visual Basic

Visual Basic

```
Private Sub CreateURLCode()
       _QRText = Trim(txtFormWebsiteAddress.Text)
      If QrCodeSizeCheck() Then
           _CodeBuilt = True
           EnableButtons()
       Else
            DispalyLengthWarning()
            Return
         End If
 End Sub
```

当输入一个URL，并单击“创建QR Code”之后，将产生以下QR二维码图像：



C1QRCode编码限制

尽管QR二维条码的白皮书描述了多大40种不同尺寸的编码，C1QRCode控件仅实现了Level1到 Level 10的尺寸。下表总结了每一种尺寸对应的图像尺寸以及支持的内容长度。

错误校验等级：L（更高的错误校验等级将减少编码容量）

Level	Size	Numeric	Alpha	**Bin**
1	21x21	41	25	17
2	25x25	77	47	32
3	29x29	127	77	53
4	33x33	187	114	78
5	37x37	255	154	78
6	41x41	322	195	134
7	45x45	370	224	154
8	49x49	461	279	192
9	53x53	552	335	230
10	57x57	652	395	271

注意：

Level 表示编码类型，可以设置为Automatic。表示自动选择编码尺寸。Size表示图像的尺寸，单位是像素。支持字母和数字，包括[0-9] [A-Z] [$ % * + /：]。

关于QR二维条码格式的详细信息，请参见：http://www.denso-wave.com/qrcode/qrstandard-e.html以及http://en.wikipedia.org/wiki/QR_code。

提高C1QRCode图像分辨率

通过设置SymbolSize属性，您可以设置构建QR二维码图像的尺寸大小。

简单地在Form设计器上选中C1QRCode控件，可以在VisualStudio属性窗体中设置SymbolSize属性的值为一个介于2和10之间的值。较大的值会导致生成较大的图像，这将占据更多的空间，但是使得扫描枪可以更容易地识别条码。

SymbolSize属性的默认值是3，它看起来像这样：

你可以设置SymbolSize属性为2和10之间的任何值。展示一下不同尺寸的示例，参见下面的图片：

SymbolSize = 2	**SymbolSize** = 10
	

BarCode for WinForms 示例

ComponentOne软件工具包配备有各种不同的示例工程以及/或演示Demo，这些示例工程可能同时使用了ComponentOneStudio套件包提供的其它的开发工具。

示例可以通过ComponentOne示例资源管理器访问。为查看示例，在桌面上单击开始按钮，并选择AllPrograms|ComponentOne|StudioforWinForms|Samples|BarCodeSamples。下表对每个示例提供了一个简短的描述。

Visual Basic 示例

示例	描述
PrintBarCodes	演示如何打印由C1BarCode控件创建的条码。此示例使用C1BarCode控件。
QRCodeConstructor	演示格式化信息如何输入并转换为一个可以保存或打印的QR二维条码。此示例使用C1QRCode控件。

C# 示例

示例	描述
BarCodeSample	演示了条码编码的不同类型。此示例使用C1BarCode控件。
PrintBarCodes	演示了怎样打印C1QRCode创建的条形码。此示例使用C1BarCode控件。