dLSoft

dBarcode.NET Components

By dLSoft

This manual was produced using *ComponentOne Doc-To-Help*.™

Contents

dBarcode.NET Components

Introduction

dBarcode.NET Components are managed code components that allows barcode images to be created within the user's own .NET application. A barcode image may be displayed on screen or printed on a printer, and the image may be passed to any other image-handling component.

Single computer version of the components are useable only on a single computer. Developer versions permit applications built with the dBarcode.NET components to be distributed up to a maximum distribution of 10,000 copies. Developer versions requires the developer's serial number to be passed to the component using the Serial property.

dBarcode.NET Components are designed to work with Visual Studio.NET and require the .NET run-time to be installed on any computer using the components. Example code is provided with each component for users of Visual Basic .NET, C# and J#.

The components

There are several components in the dBarcode.NET Components family, one for each of the most popular barcode types, and one which handle a wide range of different barcode types. The components are:

dBarcode_39.NET Component
Abcnet39Lib.dll – for Code 39, Extended Code 39, Code 93 and Extended 93 barcodes

dBarcode_128.NET Component
Abcnet28Lib.dll – for Code 128, EAN 128, EAN-14 / UCC-14 (also know as DUN-14), and SSCC (Serial Shipping Container Code) barcodes

dBarcode_EAN.NET Component
AbcnetEANLib.dll – for EAN-13, EAN-14, EAN-8, UPC-A, UPC-E barcodes

dBarcode_ITF.NET Component
AbcnetITFLib.dll – for Standard 2 of 5, Interleaved 2 of 5, and Matrix 2 of 5 barcodes

dBarcode.NET Component Universal
AbcnetLib.dll – for most common barcode types (over 30 types supported; see the list provided under CodeType).

With the exception of the CodeType property (which specifies the type of barcode to be generated) all components share the same component properties, and in this manual all components are discussed using the name of the dBarcode.NET Component. The values of the CodeType property for the individual components is described under Properties

Using dBarcode.NET Component in Visual Studio.NET

Adding a dBarcode.NET Component to the ToolBox

To add a dBarcode Component to the Visual Studio ToolBox, display the ToolBox and select the Components tab. Right click on the Components pane and select Add/Remover Items.. from the pop-up menu displayed. A dialog box is displayed listing the currently installed components. Ensure that the .NET Framework Components page is displayed.

Push the Browse button and navigate to the location where you have installed or copied your dBarcode.NET component and select the DLL (e.g. AbcnetLib.dll).

Then push the Open button.

The list of installed components is now displayed, including your dBarcode.NET component. Ensure that the checkbox alongside the component name is checked. Now push the OK button.

The component appears as an icon on the ToolBox Components pane, with one of the following class names alongside:

Component	.NET name	DLL
dBarcode.NET	Abcnet	AbcnetLib.dll
dBarcode_39.NET	Abcnet39	Abcnet39Lib.dll
dBarcode_128.NET	Abcnet28	Abcnet28Lib.dll
dBarcode_EAN.NET	Abcnetean	AbcnetEANLib.dll
dBarcode_ITF.NET	Abcnetitf	AbcnetITFLib.dll

Adding a dBarcode.NET component to a project.

With a project's form open in design mode drag the dBarcode.NET component icon from the toolbox onto the form.

The component icon appears on the panel below the form – it does NOT appear on the form itself. The instance of the component will be given a default name (eg Abcnet1) which appears in the properties panel when the component is selected. A single Form may contain any number of dBarcode.NET Components. The first to be added will be called Abcnet1, the second Abcnet2, and so on; the names may be changed by the user by modifying the Name property within the Properties box.

The properties panel also displays all other settable properties for the component, and these values will be used as defaults unless properties are changed programmatically within your project.

Setting and retrieving property values programmatically

The dBarcode.NET Components may be operated entirely by setting or retrieving Property values programmatically.

Clicking on the dBarcode Component in the panel under the form when Visual Studio's Properties box is displayed will show the current settings for component's available properties. Most of these may be edited using the Properties box, or may have their values set from within the user's program by statements of the kind

```
Abcnet1.Caption="12345"          Visual Basic
Abcnet1.Caption="1234";          C#
```

Abcnet1.set_Caption("1234"); J#

dBarcode Component properties that are set AFTER a barcode has been created may be retrieved within user's programs by statements of the kind:

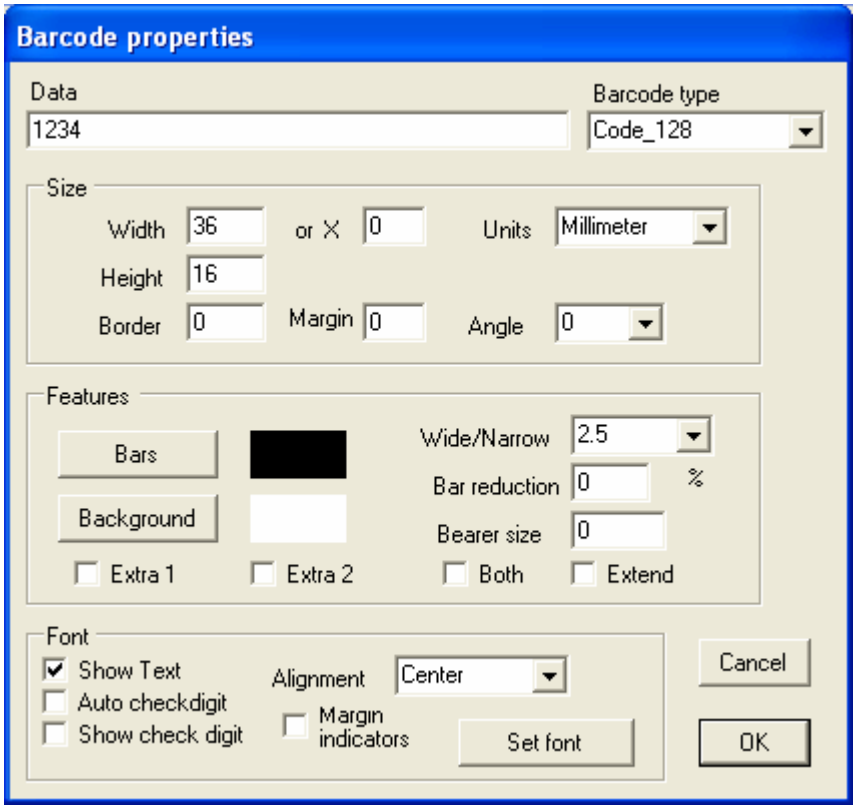
x=Abcnet1.Error Visual Basic

x=Abcnet1.Error; C#

x=Abcnet1.get_Error() J#

Setting properties through the Barcode properties dialog box

Using the Method AbcProperties() casues the Barcode properties dialog to be displayed. This displays all settable properties in a convenient form and enables changes to be made by selecting from drop-down lists or entering values into edit boxes, or summoning standard Windows dialogs for the selection of colors or fonts.



Displaying a barcode on a form

To display a barcode on a form a PictureBox is used to hold the image.

Place a PictureBox Windows Forms control on the form and add some code to your program to take the barcode image from the dBarcode.NET component and use it as the Image for the PictureBox

For example:

```
private void DoBarcode()                    Visual Basic
{
      Dim g As Graphics = PictureBox1.CreateGraphics()
      Abcnet1.Caption="1234"
      PictureBox1.Image=Abcnet1.Barcode(g)
      g.Dispose()
}

private void DoBarcode()                    C#
{
      Graphics g=pictureBox1.CreateGraphics();
      abcnet391.Caption="1234";
      pictureBox1.Image=abcnet391.Barcode(g);
      g.Dispose();
}

private void DoBarcode()
{
      Graphics g=pictureBox1.CreateGraphics();
      abcnet391.set_Caption("1234");
      pictureBox1.set_Image(abcnet391.Barcode(g));
      g.Dispose();
}
```

Printing a barcode image

Printing the image returned by the Barcode() call may be accomplished by any of the printing techniques available for Visual Studio.NET project. However, probably the most useful approach is to use the DrawImage() method in a PrintPage handler as illustrated below, and in the examples provided with the components:

Visual Basic

```
Private Sub PrintDocument1_PrintPage(ByVal sender As System.Object, ByVal e As
System.Drawing.Printing.PrintPageEventArgs) Handles PrintDocument1.PrintPage

   e.Graphics.PageUnit = GraphicsUnit.Document '
   e.Graphics.DrawImage(Abcnet1.Barcode(e.Graphics), 300.0F, 300.0F)
   e.HasMorePages = False
End Sub
```

C#

```
private void printDocument1_PrintPage(object sender,
System.Drawing.Printing.PrintPageEventArgs e)
{
      e.Graphics.PageUnit = GraphicsUnit.Document;
      e.Graphics.DrawImage(abcnet1.Barcode(e.Graphics), 500F,500F);
      e.HasMorePages = false;
}
```

J#

```
private void pd_PrintPage(Object sender,
System.Drawing.Printing.PrintPageEventArgs e)
{
      e.get_Graphics().set_PageUnit(GraphicsUnit.Document);
      e.get_Graphics().DrawImage(abcnet1.Barcode(e.get_Graphics()),500F, 500F);
      e.set_HasMorePages(false);
}
```

While the PageUnit setting can be any of the allowed values, we have found that the most accurately sized barcodes are produced when the highest resolution setting (Document, equivalent to 300 units per inch) is used.

Licensing the component

dBarcode.NET components will behave as Testware versions (generating deface barcode and truncated pattern strings) unless correctly licensed.

Single computer version of the components are licensed for use only on a single computer. Developer versions permit applications built with the dBarcode.NET components to be distributed up to a maximum distribution of 10,000 copies.

Developer versions require the developer's serial number to be passed to the component using the Serial property before the Barcode() method is called. The Serial property should be set to the serial number provided for your product, e.g.

Abcnet1.Serial="12345" for VB

abcnet1.Serial="12345"; for C#

abcnet1.set_Serial("12345"); for J#

Reference

Summary of Properties and Methods

These most important properties and methods are outlined below, and followed by sections in which the properties and methods are defined in detail.

Component properties.

The properties that can be set to generate an image

Abcnet1.AutoCheckdigit specifies whether any check digits are calculated automatically

Abcnet1.BackColor the colour behind the bars

Abcnet1.BarRatio specifies the ratio of wide/narrow bar for some barcode types

Abcnet1.BarcodeHeight required target height of barcode image (in units determined by the setting of the Unit property)

Abcnet1.BarcodeWidth required target width of barcode image (in units determined by the setting of the Unit property)

Abcnet1.BearerSize specifies the thickness of bearer bars for those barcodes that may have bearer bars

Abcnet1.BorderWidth specifies the thickness of any border around the image

Abcnet1.BothBearers determines whether both upper and lower bearer bars are displayed

Abcnet1.Caption specifies the characters that make up the code

Abcnet1.CodeType specifies the barcode type required

Abcnet1.ExtendBearers Allows bearer bars to extend into light margins.

Abcnet1.ForeColor the colour of the bars and any text under the bars

Abcnet1.Font The font used to render any text under the barcode

Abcnet1.Indicators specifies whether light margin indicators should be displayed for those barcode types that support these. Not that the prefix and suffix digits of UPC barcodes and the prefix digit of EAN-13 and EAN-8 barcodes are regarded as light margin indicators for this purpose.

Abcnet1.MarginSize specifies the size of the light margins

Abcnet1.Orientation specifies the orientation angle of the barcode image

Abcnet1. Reduction specifies the percentage reduction in bar thickness (useful for allowing for ink spread in wet-ink printing processes.

Abcnet1.ShowText specifies that the text content of the barcode should be displayed under the bars

Axbarcode.ShowCheckdigit specifies whether any automatic check digit is displayed (for those barcode type which permit this)

Abcnet1.TextAlign specifies the text justification for human readable text under the barcode

Abcnet1.Xunit specifies the thickness of each barcode element in mils (1/1000 inches)

Many of these properties have default values (see the reference section), so do not require changing if you can make do with the default values. The properties that must be set for you to obtain a barcode are

Abcnet1.CodeType specifies the barcode type required

Abcnet1.Caption specifies the characters that make up the code

A typical example of setting these properties is:

Abcnet1.CodeType=Code_39

Abcnet1.Caption="123456"

The following Information properties are available after calls to the Barcode() method.

Abcnet1.Error an error code which is non zero if an error has occurred in generating the barcode image.

Abcnet1.ImageHeight the height of barcode picture generated (in units determined by the setting of the PageUnit property of the Graphics object passed into the component)

Abcnet1.ImageWidth the width of barcode picture generated (in units determined by the setting of the PageUnit property of the Graphics object passed into the component)

Abcnet1.Pattern a pattern string containing 1s and 0s representing the bars and spaces respectively for the barcode symbol (see Pattern details for clocked code variants).

Abcnet1.Status a string interpretting the error code produced when generating the barcode image.

Abcnet1.String2 the barcode content in text form (including any automatically calculated check digit).

Component methods

Barcode(Graphics g) causes a barcode image to be created using the Graphics properties of g

Properties() Displays the Barcode properties dialog for the component.

dBarcode.NET Component Properties

dBarcode.NET Component properties fall into two categories - those that are required to specify characteristics of the barcode image, and those that provide information about the barcode.

Required properties

The following properties are required - although all are provided with default values.

AutoCheckdigit

Type: BOOL

Default: FALSE

Allowed values: FALSE (checkdigit characters not calculated)

TRUE (checkdigit characters calculated and appended to code for appropriate code types)

BackColor

Type: Color

Default: Color.White

Allowed values: any allowed Color value.

Sets the colour of the image background. This value may be over-ridden by the Transparent property.

BarRatio

Type: float

Default 2.5

Allowed values: 2.0 – 3.0

This setting allows some barcode types to have the Wide bar/Narrow bar ratio modified.

Applies mainly to Code 39 and Interleaved 2 of 5 barcodes.

BearerSize

Type: float

Default: 0

Allowed values: 0 - 10000

Allows the thickness of bearer bars for those barcodes which support bearers to be set (in units determined by the setting of the PageUnit property of the Graphics object passed into the component).

Note that for most barcode types the number of bearer bars displayed depends on the setting of the BothBearers property. If BothBearers is True the bars are displayed above and below the barcode. If BothBearers is False then only a single bearer bar is displayed above the barcode.

BorderSize

Type: float

Default: 0.0 (no border)

Allowed values: any

Specifies the size (in units determined by the setting of the PageUnit property of the Graphics object passed into the component) of a border placed around the barcode. The border area is created using the background color set by the BackColor property.

BothBearers

Type: BOOL

Default: FALSE

Allowed values: TRUE or FALSE

When TRUE causes bearer bars to be draw above and below the barcode. When FALSE no bearer bars are drawn if the BearerSize property is zero. Otherwise a FALSE setting causing only a single bearer bar to be displayed above the barcode.

Caption

Type: string

Default: "12345"

Allowed values: Any text string.

Note: only text strings recognised as valid barcodes will result in a barcode picture. An Illegal character in the text string will cause an Error value to be set.

CodeType

Type: bCode enumeration member

For dBarcode.NET Component Universal

Default: Code_39

Allowed values: The ranges of values defined for the individual components are shown in the barcode type tables below.

The barcode type can be set using either the CodeType property or the CodeTypeValue property

CodeTypeValue

Type: integer

Default: 0

Allowed values: The ranges of values defined for the individual components are shown in the barcode type tables below.

The barcode type can be set using either the CodeType property or the CodeTypeValue property

For dBarcode.NET Universal

Barcode	**CodeType**	**CodeTypeValue**
Code 39	Code_39	0
Extended Code 39	Extended_39	1
Codabar	Codabar	2
2 of 5	Standard_2_of_5	3
Interleaved 2 of 5	Interleaved_2_of_5	4
Matrix 2 of 5	Matrix_2_of_5	5
Code 93	Code_93	6
Extended Code 93	Extended_93	7
Code 128	Code_128	8
EAN 128	EAN_128	9
SSCC	SSCC	10
EAN 14	EAN_14	11
EAN 13	EAN_13	12
EAN 13 + 2 digits	EAN_13_plus_2	13

EAN 13 + 5 digits	EAN_13_plus_5	14
EAN 8	EAN_8	15
EAN 8 + 2 digits	EAN_8_plus_2	16
EAN 8 + 5 digits	EAN_8_plus_5	17
UPC-A	UPC_A	18
UPC-A + 2 digits	UPC_A_plus_2	19
UPC-A + 5 digits	UPC_A_plus_5	20
UPC-E	UPC_E	21
MSI/Plessey	MSI_Plessey	22
Plessey	Plessey	23
DeutschenPost	DeutschePost	24
PostNet	PostNet	25
Planet Origin	Planet_Origin	26
Planet Destination	Planet_Dest	27
Royal Mail	RM4SCC	28
4-State	FourState	29
ISBN	ISBN	30
ISSN	ISSN	31
ISMN	ISMN	32

For dBarcode.NET 39

Barcode	**CodeType**	**CodeTypeValue**
Code 39	Code_39	0
Extended Code 39	Extended_39	1
Code 93	Code_93	2
Extended Code 93	Extended_93	3

For dBarcode.NET 128

Barcode	CodeType	CodeTypeValue
Code 128	Code_128	0
EAN 128	EAN_128	1
SSCC	SSCC	2
EAN 14	EAN_14	3

For dBarcode.NET EAN/UPC

Barcode	CodeType	CodeTypeValue
EAN 128	EAN_128	0
EAN 14	EAN_14	1
EAN 13	EAN_13	2
EAN 13 + 2 digits	EAN_13_plus_2	3
EAN 13 + 5 digits	EAN_13_plus_5	4
EAN 8	EAN_8	5
EAN 8 + 2 digits	EAN_8_plus_2	6
EAN 8 + 5 digits	EAN_8_plus_5	7
UPC-A	UPC_A	8
UPC-A + 2 digits	UPC_A_plus_2	9
UPC-A + 5 digits	UPC_A_plus_5	10
UPC-E	UPC_E	11

For dBarcode.NET ITF

Barcode	CodeType	CodeTypeValue
2 of 5	Standard_2_of_5	0
Interleaved 2 of 5	Interleaved_2_of_5	1
Matrix 2 of 5	Matrix_2_of_5	2

ExtendBearers

Type: BOOL

Default: FALSE

When TRUE allows bearer bars to extend into light margins. When FALSE bearer bars cover the bars only.

Extra1

Type: BOOL

Default FALSE

These additional properties are not normally used. However, they do provide additional functions for a limited number of specific barcode types. See Barcodes section for details.

Extra2

Type: BOOL

Default FALSE

These additional properties are not normally used. However, they do provide additional functions for a limited number of specific barcode types. See Barcodes section for details.

Font

Type: FONT

Default: Arial 10 point

Allowed values: Any accessible TrueType font.

ForeColor

Type: Color

Default: Color.Black

Allowed values: any valid Color

Sets the colour of the image foreground, i.e. the bars and text colour.

Indicators

Type: BOOL

Default: FALSE

A value of TRUE causes the light margin indicators to be displayed. For some EAN barcodes there are recommended ways for the margin indicators to be shown on the image, and the Prefix code and checkdigit for UPC-A and UPC-E codetype to be displayed in the light margins. A value of FALSE prevents the display of the light margin indicators.

MarginSize

Type: FLOAT

Default: 0.0

Allowed values: any

The MarginSize property sets the Light Margin space on either side of a barcode image. The units are the units specified by the current ScaleMode. (in units determined by the setting of the PageUnit property of the Graphics object passed into the component)

Orientation

Type: integer

Default: 0 The value of this parameter determines the orientation of the barcode image created.

Allowed values: 0 - 360 degrees

Note that the rotation of text is only supported by TrueType and other rotatable fonts. Note also that some applications do not correctly handle metafiles that contain rotated text.

Reduction

Type: integer

Default: 0 the thickness of each line drawn on the barcode image is reduced by this percentage amount. This property may be used to compensate for ink spreading during wet-ink printing.

Allowed values: 0 - 50 (%)

A positive reduces the thickness of bars by the calculated amount of the thinnest bar. A negative value reduces the thickness of bars by the percentage applied to the bar thickness (so a bar three units wide is reduced by an absolute amount that is three times greater than that which would apply to a one unit bar)

ShowCheckdigit

Type: BOOL

Default: FALSE

When set to TRUE this property causes the any automatically calculated check digit to be included in the any text displayed along with the barcode. When set to FALSE the check digit is not displayed.

Note that this property has no effect on those codetypes for which the checkdigit display is mandatory (including EAN and UPC codes), or for which the check digit is never displayed (Code 128 and EAN 128).

ShowText

Type: BOOL

Default: TRUE

When FALSE text version of the code is NOT included in the barcode image. When TRUE text version of the code IS included in the barcode image.

This property is ignored for clocked codes which have no text version.

TextAlign

Type: `StringAlignment`

Default: Center

Allowed values: Near, Center, Far

TextAlign sets the alignment of any text displayed under the barcode. A value of Center centers the text, Near gives left justification and Far gives right justification. Justification is to the edge of the barcode – NOT the edge of the light margins.

Unit

Type GraphicsUnit

Default: Millimeter

The scaling mode applied to values of the following size-related properties:

BarcodeHeight
BarcodeWidth
BearerSize
BorderWidth
MarginSize

Allowed values:

Document
Display
Pixel
Point
Millimeter
Inch

Xunit

Type: float

Default: 0.0

Allowed range: 8.0 – 255.0

The Xunit property may be used to specify the width (in Mils) of the smallest element in the barcode.

Note that setting this property to a value other than 0 causes the image to resize itself to a width calculated from the number of X units in the barcodes – so the value of the BarcodeWidth property is ignored.

Using values smaller than 8 will produce a barcode image, but that image will not meet standard specifications and may not scan.

Information properties

Note that the barcode symbol is determined onlywhen this Barcode() method is called, so the information properties are not available until this method has been called.

Error

Type: integer

Returns a value representing the error code if a valid barcode image cannot be created. Otherwise returns 0.

Read only. Do not set this property.

The error codes and corresponding Status property values are shown below:

Status

Type: string

Returns a string interpretting the value of the Error property.

The error codes and corresponding Status property values are shown below:

Error	Status
0	OK
1	Illegal character in data
2	Wrong data length
3	Error in barcode data

Read only. Do not set this property.

ImageHeight

Type: FLOAT

Return the actual height of the image generated, in units of the current Graphics object – NOT the units specified by the Unit property.

The value returned includes the size of any border, margin and text.

Read only. Do not set this property.

ImageWidth

Type: FLOAT

Return the actual width of the image generated, in units of the current Graphics object – NOT the units specified by the Unit property.

The value returned includes the size of any border, margin and text.

Read only. Do not set this property.

Pattern

Type: String

Returns: A string containing a pattern of 0s and 1s, where 0 represents a space of minimum width and a 1 represents a bar of minimum width. The 1s may be used to indicate that a bar (filled rectangle) must be drawn.

Read only. Do not set this property.

String2

Type: string

Returns the Caption property plus any additional characters generated as check digits.

Read only. Do not set this property.

Methods

Barcode(Graphics g)

Type: Metafile

This method causes a barcode image to be created using the Graphics properties of g (including the PageUnit property which determines that absolute size information embedded in the image).

Note that the barcode symbol is determined onlywhen this method is called, so the information properties are not set until this method has been called.

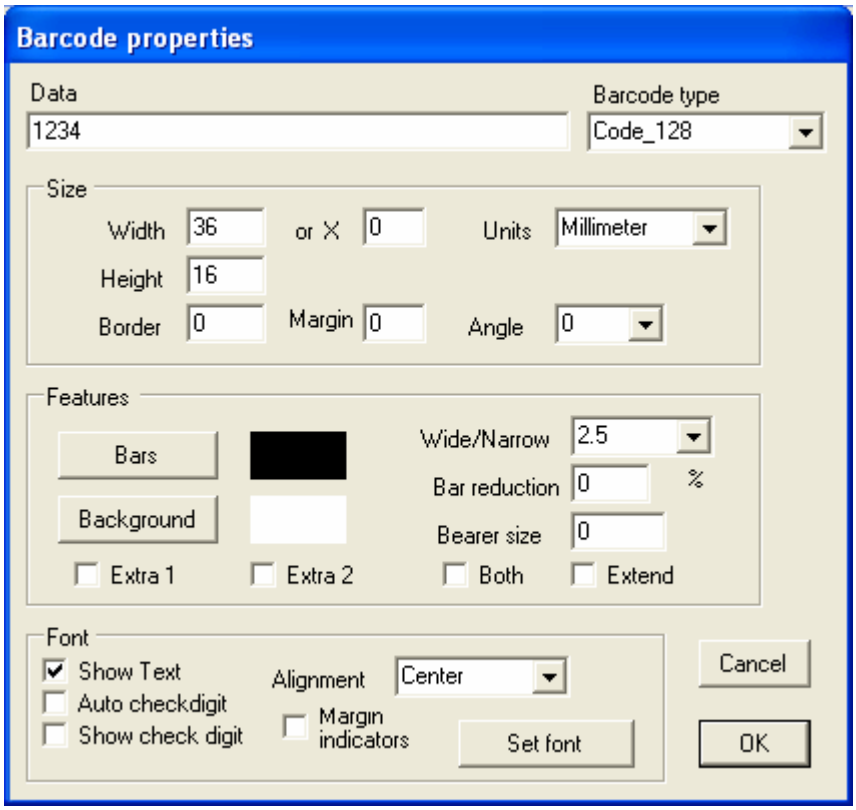
Properties()

Type: BOOL

Displays the Barcode properties dialog for the component.

Returns TRUE if the user exits the dialog by pushing the OK button and sets all component properties to those specified in the dialog.

Returns FALSE if the user exits the dialog by pushing the Cancel button.



Barcodes

Several fundamental characteristics of barcodes need to be understood by users of dLSoft barcode products:

1. The thickness of bars in barcodes is important. The size of the smallest element of a barcode is know as its X unit size or X dimension and in standards this is usually specified in units of Mils (0.001 inches).

dLSoft barcode products may refuse to create a barcode image if the bar thickness within the metafile becomes too small. However, even when dLSoft barcode products creates an image you may resize it within another application so that when it is printed by the other application its lines may too small for the printer's resolution. Consequently it is essential that you check that a printed barcode is readable using an appropriate scanner or reader.

Barcodes printed by laser printer will, in general, be printed correctly, but codes printed by matrix printers must be reproduced at a large enough scale that the barcodes unit size is at least as large as the printer's pins.

Bar reduction: All dLSoft barcode products allow the thickness of bars to be reduced (for example to allow for ink spread during wet ink printing processes), but this adjustment should only be made when the knowledge of the extent of reduction required is available. Random guesses usually produce unreadable images!

2. Many barcode types may use codes only of a specific length. (e.g. EAN13 requires 13 digits in the code). Some barcode type use specific digits of the code as a checksum - so not every combination of digits can form a legal barcode. dLSoft barcode products can optionally calculate checksum digits, requiring only the other digits to be entered by the user. Furthermore most coding schemes are limited to 32 characters or less.

3. The barcode types supported in this release are shown in the barcodes table shown under CodeType. If you plan to use a specific barcode type you should examine the notes on that type before printing any barcode images.

4. Users should be aware that it is possible to generate barcodes of a specific type and find that normal retail scanners are unable to decode the images. This does not necessarily mean that there is anything wrong with the barcode image. Most scanners aimed at the retail market are not programmed to interpret barcode codes reserved for other (eg. military) use.

5. The Extra options. All of our products provide access to two barcode-specific options. These are the options EXTRA1 and EXTRA2, which appears as checkboxes in the Barcode properties dialog, or may be set as Boolean properties. These options are used only for a limited number of barcodes, which have “unusual” features. The effect of these options is described under the barcode types, which use them. For all other barcode types these options may be ignored or set to false.

Barcode types

EAN-13

EAN-13 is the main scheme used throughout Europe for retail article numbering. It is a numeric only coding scheme. The > symbol in the right margin is a light margin indicator. In the left margin the first code digit is used as the margin indicator. No other marking should appear in the light margins.

EAN-13

EAN codes require 13 digits (12 if the check digit is calculated automatically. Numbers used for EAN article numbering are assigned by the country's Article Number Association (the ANA in the UK).

EAN codes may contain 2 or 5 digit supplementaries:

EAN13+2

EAN13+5

The JAN coding scheme is EAN13 with the first two digits being 49.

Note that there is not a one to one correspondence between bars and the code numbers.

EAN-8 is a smaller and shortened version of the EAN code.

EAN-8

EAN-8 requires 8 digits (7 if the check digit is calculated automatically), and support 2 and 5 digit supplementaries.

ISBN

The ISBN coding scheme is EAN-13, with the first three digits being 978, and 9 digits the ISBN number of the book (without check digit). The final digit is the EAN calculated check digit.

ISBN

Note that the final digit of a 10 digit ISBN number is an ISBN check digit and this is NOT included in the barcode image. The barcode image will contain the EAN check digit.

Two or five digit supplementary characters may be added to ISBN codes by placing the supplementary after a / character.

Bookland barcodes are unique numbers that are printed on the covers of books. They contain the book's ISBN number and pricing information encoded using EAN 13 bar codes with a 5 digit supplementary code.

For a book with **ISBN 1-234-5678-9** retailing at **$19.95** in the US, the data to encode is generated by taking **978**, followed by the ISBN number 123456789 *(the last digit of the ISBN number is a check digit and is not included*), followed by a currency digit (5 for US$) and a four digit price (51995), ie.

97812345678951995

This may be passed to dBarcode as 1-1234-56789/51995

The spacing of the text above the barcode may be modified by entering a character spacing value between 50 and 100%.

ISSN

The ISSN coding scheme is EAN13, with the first three digits being 977, 7 digits showing the ISSN number of the periodical (without check digit), and 2 spare digits (used in the UK to indicate price code changes). The final digit is the EAN calculated check digit.

ISSN

Two or five digit supplementary characters may be added to ISBN codes by placing the supplementary after a / character.

ISMN

The ISMN coding scheme is EAN13, with the first four digits being 9790, followed by the first 8 digits of the assigned Music number. The Music number check digit is not required. The final digit in the barcode is the EAN check digit.

JAN

The JAN coding scheme is then same as EAN13 with the first two digits being 49.

Codabar

The Codabar coding scheme is a self-checking system which has 16 characters in its character set; the digits 0 -9, and the characters $: / . + -. It has a choice of four start & stop characters, although some versions allow a choice of eight!. By default dLSoft barcode products uses A and C for start and stop respectively. However, by prefixing the barcode with a caret (^) and two symbols, any of the allowed Codabar characters may be used for start and stop; ie.

^AT

causes A to be used as the start character and T to be used for the stop character.

Codabar

The allowed Codabar start and stop characters are: A B C D E N T *

The start and stop characters are not displayed in text form.

Matrix 2/5

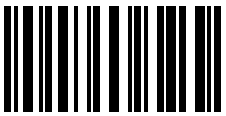
Matrix 2/5 is an older numeric-only code, with an optional modulo 10 check digit. Not recommended for new applications.

Matrix 2/5

Code 128 & EAN128

Code 128 and EAN-128 are modern very high density coding schemes. They have three coding schemes each and permit the inclusion of special characters not present on the keyboard. If no coding scheme is specified scheme B is used by default unless the automatic conversion option (see below) is chosen. For EAN-128 scheme C is used for any code, which has numbers in the first four digits (as recommended by the EAN). An alternative scheme may be selected by making the first character one of the start characters specified below.

EAN-128



Code-128

The special characters may be entered as <ALT>0XYZ, where XYZ is the 3 digit ASCII code (+128 for values <32), or according to the following table:

XYZ	character	Code A	Code B	Code C
197	Å	DEL		
198	Æ	func. 3	func. 3	
199	Ç	func. 2	func. 2	
200	È	shift	shift	
201	É	code C	code C	
202	Ê	code B	func. 4	code B
203	Ë	func. 4	code A	code A
204	Ì	func. 1	func. 1	func. 1
205	Í	Start A	Start A	Start A
206	Î	Start B	Start B	Start B
207	Ï	Start C	Start C	Start C
208	Ð	NUL		

Code C codes only the digit pairs 00-99.

Note that EAN-128 codes have parentheses removed before coding, so (and) may appear in the human readable form but will be omitted from the barcode. Parentheses may not be used as part of the code data.

Spaces may be stripped from the text provided for input by checking the EXTRA1 checkbox in the Barcode properties dialog, or setting the Extra1 to TRUE. This allows spaces to appear in the text under the symbol while not being included in the symbol itself.

dBarcode.NET components normally provide the control codes for switching between subtypes automatically, but this facility can be turned off by setting the EXTRA2 property to TRUE. When dBarcode.NET is to provide control codes no additional control codes should be provided by the user, although an initial Start A or Start B code may be given if it is desired to force the symbol to start in a particular code type. Users should note that if this option is chosen then the barcode produced may not appear identical to a sample obtained from another source - although it will scan to produce the same characters.

EAN128 defines the use of Application Identifiers (AIs) – which are numbers with a predefined meaning and usually enclosed in brackets in the human readable form. While many AIs are followed by fixed length strings, some may be followed by a variable length string – in which case the string is terminated with a Function 1 character.

The majority of support calls result from users not using the correct 128 code variants (ie. A, B or C) or not being aware of which code variant a customer is expecting. Some customers expect only Code C, while others start in Code A and then switch to Code C, etc. It is important to be aware that the three code variants exist and will commonly be encountered within the same barcode. For this reason it is essential to ascertain which type the customer wants and if and where the code variant should change along the barcode.

Users of Code 128/EAN 128 should note that while there is a nominal size for these symbols (31.8 mm high and 11*n+2 mm long, where n is the number of characters including control codes), many applications of these codes use recommended sizes of between 50% and 84% of nominal.

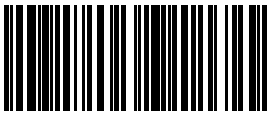
DUN-14

DUN-14 is an older name for the EAN-14 barcode type.

EAN-14

EAN-14 barcodes may be represented by ITF or EAN128 barcodes, and modern implementations should use EAN128 – so that is what dBarcode uses. EAN-14 barcodes may be constructed from 12/13 digit retail UPC/EAN barcode numbers by left-filling the numbers with zeros, and uses a special checkdigit.

EAN-14 in EAN128 barcode form may be created by providing 13 digits; the Logistical Variant digit (normally 0 in the UK) followed by the first 12 of the retail digits (eg. from EAN-13 numbers but without EAN's check digit). Creation from UPC-A numbers requires two 0s followed by the first 11 digits of the UPC-A barcode number without the check digit. dBarcode calculates the EAN-14 check digit if Auto-checkdigit is enabled, and then produces the EAN128 barcode.



If the EAN-14 checkdigit is to be provided the 14 digits are required, and Auto-checkdigit should be disabled.

SSCC

The Serial Shipping Container Code is a unique identification of individual shipping containers. The standard includes a unique barcode symbology, UCC/EAN-128, using the UCC/EAN Application Identifier Standard.

The SSCC uses an 18 digit number which consists of:

a) a single extension digit assigned by the company that constructs the SSCC

b) the UCC/EAN company prefix. Those assigned by UCC are prefixed with 0.

c) a serial reference number that must remain unique for at least 12 months

d) a single Mod 10 check digit.

When an SSCC barcode is generated using dLSoft components the data is prefixed by the (00) Application Identifier. The Mod 10 check digit may be generated by selecting the Auto-check digit option. The Show check digit option is ignored, as the Mod 10 check digit must always be shown in the human readable form.

UPC

The UPC (Universal Product Code) is widely used in the USA as a retail code. However, it has wider application and this can result in some confusion.

The actual UPC code is a 10 digit code. The 10 digit number is preceded by a “number system” digit, which is 0 for the retail version, and followed by a check digit. In many retail systems only the 10 digits of the UPC code need to be entered in the event of a mis-scan, so there have been times when the leading 0 has not been included in the human readable form. However, other values of the number system digit are used for specific purposes (eg. 6 or 7 are used for manufacturing identification numbering, 3 for drug products, etc.).

The UPC-A code is one variant of a number of 12 digit codes widely used in the USA. Retail codes are usually thought of as those with 10 digits (or 11 if the checkdigit is being entered explicitly), and in fact are 12 digit codes made up of a leading 0, followed by 10 product digits and 1 checkdigit.

The library generates the barcode images for UPC-A if the leading 0 is provided, followed by the 10 digit product code. The check digit may either be entered explicitly or calculated by the library. This technique allows alternative leading digits to be used for their intended purposes. Users of such alternative codes will know what those leading digits may be, or may obtain the information from the authorised code provider.

The UPCC has produced more than one specification of the UPC codes. The current specification suggests that the country code (always 0 in the USA) and the codes checkdigit should be printed aligned with the coded digits, but in the light margins. Earlier specifications suggested that these digits should be printed in different positions or not at all.

The library offers the choice of not printing the digits or of printing them in the light margins (using the Indicators =TRUE option) for both UPC-A and UPC-E codes.

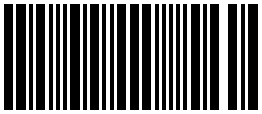
UPC-A

UPC-E

UPC-A codes support 2 and 5 digit supplementaries.

2 of 5

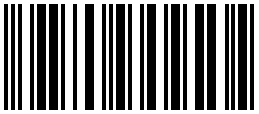
2 of 5 is a numeric only coding scheme, which is not very efficient and not recommended for new applications.



2 of 5

Interleaved 2 of 5

One of the most common codes outside the retail area is Interleaved 2 of 5, a high density, continuous numeric symbology that codes digit pairs. Because of this I-2 of 5 can only be used for even numbers of digits. If an odd number of digits is used in the DLSoft library a leading 0 is added automatically to the front of the number.



Interleaved 2 of 5

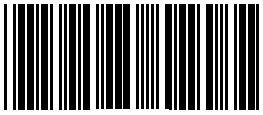
Deutschen Post

The I-2 of 5 barcode also forms the basis of the German Identcode and Leitcode symbols used by Deutschen Post. These are 12 and 14 digit barcodes respectively, but they do use a different check digit calculation from the standard I-2 of 5 symbol.

Note that spaces and periods are removed from the strings supplied for I-2 of 5 barcodes before the barcode image is created – so the correct layout for Identcode and Leitcode text may be used to create the symbols.

Code 39

Code 39 is by far the most common barcode scheme outside the retail area and is read by most scanners, although it is not as compact as Code 93 or Code 128. The normal Code 39 scheme encodes both numbers and upper case letters, and was the first alphanumeric symbology:



Standard Code 39

Code 39 has an optional checkdigit.

The Extended Code 39 scheme also includes the lower case letters and much punctuation.

Extended Code 39

It should be noted that Extended Code 39 represents most of the additional characters by using two characters from the standard Code 39 character set. Consequently Extended Code 39 symbols are about twice as long as standard Code 39 symbols.

Code 39 is a discrete symbology - so the gap between ciphers may be larger than a unit space. Some users mistake that inter-cipher gap for a space and become concerned because it is not the same size as in another barcode representing the same characters. There is no substitute for testing the barcode with a scanner!

Also the start and stop characters are the same, and sometimes may be represented in the human readable form by an asterisk.

If iExtra1 is set the start and stop characters are shown as an asterisk in the human-readable form.

If iExtra1 is not set then the start and stop characters are not represented in human-readable form.

Code 93

Code 93 was designed to complement Code 39 and is a more compact code than the latter. The library supports both the standard Code 93 (numbers and upper-case letters) and the Extended (full ASCII) Code 93.

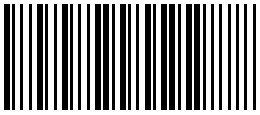
Code 93

MSI

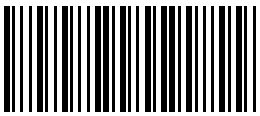
MSI, also known as the Modified Plessey Code, is a relatively weak code that is inefficient in use of space, but still used in the grocery sector.

MSI with single checkdigit

Normally this code has a single Modulo 10 check digit. However, there are two variations of a double check digit form in common use. One uses a Mod 11 check digit before the normal Mod 10 check digit, the other uses two Mod 10 check digits.



MSI with extra Mod 10 checkdigit



MSI with extra Mod 11 checkdigit

These two-checkdigit forms are accessible through the use of the Extra1 or Extra2 parameters. The effects are as shown below. Note that BOTH options also require the autocheckdigit calculation to be enabled.

If Extra1 is TRUE then a Modulo 10 check digit is calculated and inserted before the normal checkdigit.

If Extra2 is TRUE then a Modulo 11 check digit is calculated and inserted before the normal checkdigit.

Some scanning equipment cannot read both forms. (In fact some scanning equipment cannot read either of the two checkdigit forms). Check your scanners documentation to ensure that you choose an appropriate combination. DO NOT SET BOTH Extra1 and Extra2 to TRUE.

PostNet

PostNet codes are the clocked codes used in the US mail system. There are three types of PostNet code (identified as A, C and C'), which differ in the number of characters encoded. These codes are based on the US ZIP code system.

PostNet A

The allowed lengths of the PostNet data are 5, 9 and 11 characters. Any other Caption length will be reported as an error.

Planet

Planet codes are clocked codes used within the US Postal Service for the confirmation of incoming or outgoing mail.

Planet_OC

Planet codes consist of 9 digits prefixed by a code to indicate whether the Origin Confirm or Destination Confirm service is required, and postfixed by a mandatory check digit.

RM4SCC

RM4SCC is the Royal Mail (UK) version of the 4 State clocked barcode used for directing mail. The codes contain a start and stop bit, while the 4 State code (below) does not. While both codes offer the option of a checkdigit, it should be noted that the Royal Mail code must include the checkdigit (which should be calculated automatically).

These codes are based on the UK Post Code system, but may also contain an International Prefix and a Delivery Point Suffix.

Note that in both RM4SCC and 4 State (see below) all characters are converted to upper case prior to encoding and any illegal characters with ASCII codes >32 are converted to X. Illegal characters with ASCII codes <= 32 are ignored -- so spaces and carriage returns are ignored.

RM4SCC

4 State

4 State is similar to the RM4SCC code and is used in some European countries without the start and stop bits, and in some cases without the Checkdigit. This code is referred to as 4 State.

4 State

Plessey

An older code still popular in some industries, the Plessey code supports numbers and the characters X, B, C, D, E and F, plus a two character crc check.

Plessey

It is common practice in some industries using Plessey barcodes to separate the barcode characters from their checkdigits. This can be done by setting the EXTRA2 property to TRUE, or by manually including a space character at the end of the barcode data.

Location numbering

With effect from March 1st 1995, EAN International has agreed that all numbering organisations will standardise on the product numbering (ie standard EAN-13) check digit algorithm when calculating check digits for location numbers. The cut-off date for using older check digit algorithms is January 1st 1997.

When producing location numbering barcodes of the EAN-13 pattern users should check whether the newly recommended check digit algorithm (ie that produced by dBarcode) is suitable for their purposes.

Index

R

S

T

U

X