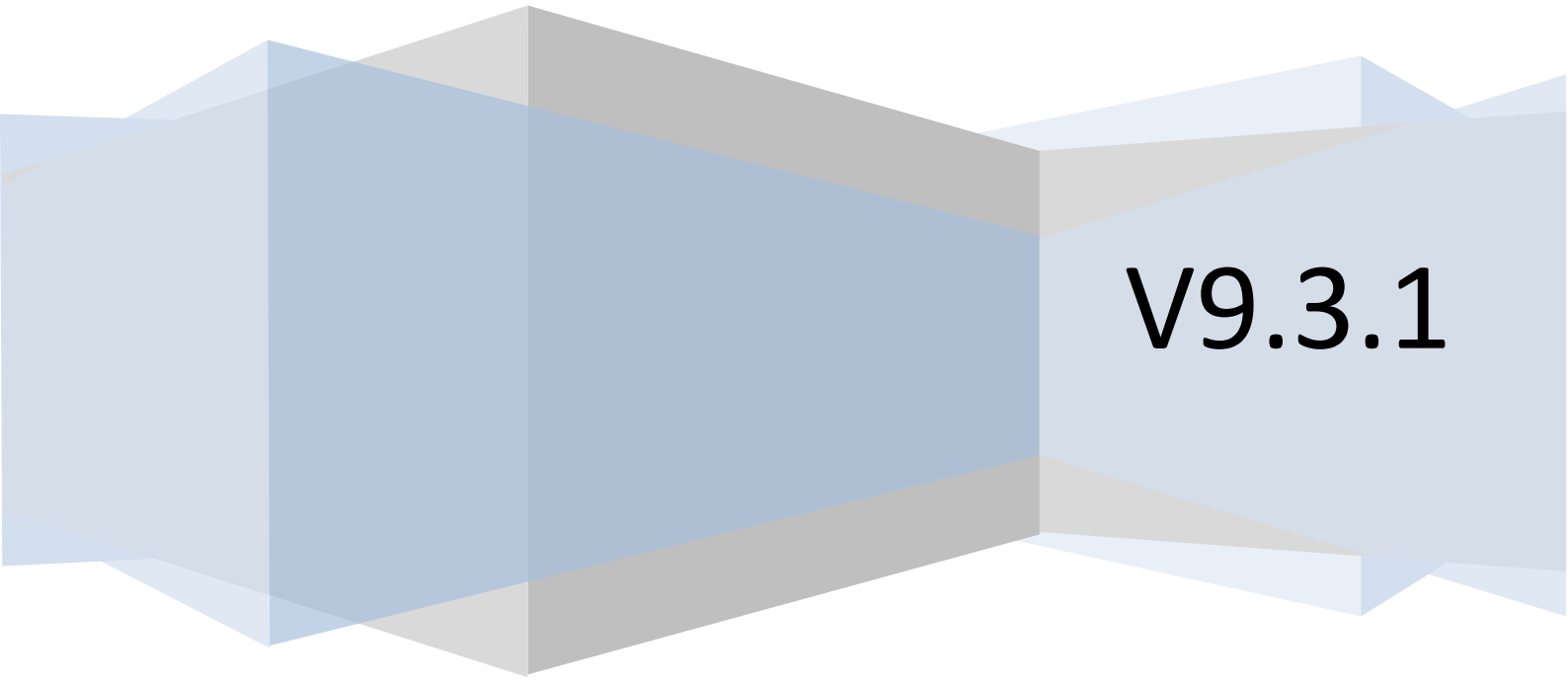
Softek Software Ltd

Softek Barcode Reader Toolkit

Product Documentation (including the PDF Extension)



V9.3.1

1 Contents

2 Overview

The Softek Barcode Reader Toolkit is an SDK for reading barcodes from images and is available in editions for Windows, Linux (including OSX and other similar platforms) or as a multi-platform Nuget package (Windows, Linux and OSX).

All barcode reading takes place via a single DLL or shared object file. On Windows it is called SoftekBarcode.DLL or SoftekBarcode64.dll and on Linux it is called libbardecode.so.

There are various ready made interfaces to the toolkit which wrap around the main barcode reading DLL or shared object. The download for the product includes everything necessary for using all the features of the toolkit.

The section on licensing contains a table that cross references features against the different licenses.

3 Download packages

3.1 .NET

The .NET interface for the SDK can be installed as a NuGet package either for .Net Standard or for .NET Framework (where a .NET Standard package can't be used).

The URL for the .NET Standard package is:

https://www.nuget.org/packages/softekbarcodenetstandard

The package supports Windows, Linux and OSX host platforms and can either be used with licenses for the Windows or Linux versions of the SDK or under a multi-platform license.

The URL for the .NET Framwork package is:

https://www.nuget.org/packages/SoftekBarcodeNet

For more information on these interfaces please refer to the section on Using the Toolkit with .NET.

3.2 Windows

The Windows download is available as a zip or self extracting exe. The main DLL files can be found in the folder called "DLL". There is also a demo program (SoftekSDKDemo), sample images and folders containing wrappers for various interfaces to the toolkit.

The URL for downloading the Windows SDK is:

https://www.bardecode.com/en1/download/

3.3 Linux, OSX etc

The Linux edition is normally downloaded as a tar ball. Extract the contents to a folder and then run:

./configure.sh

Select the most appropriate build of the toolkit for the platform being used. Note that there may be multiple options available depending on the platform. If there is a build available then the configure tool will create a link from the 'arch' folder to the correct build. Other builds for the toolkit can be found under the 'build' folder.

The 'bin' folder will contain a shell tool called bardecode and the 'lib' folder will contain the static and shared object library files. There is also a folder called 'slim' which contains a reduced size version of the library that only supports reading from bitmaps held in memory. The other folders contain wrappers for various interfaces to the toolkit.

The URL for downloading the Linux/OSX SDK is:

https://www.bardecode.com/en1/download/

4 Using the toolkit

Code using the toolkit generally has the following strcuture:

- Create an instance of the toolkit
- Set the license key
- Set properties
- Scan an image
- Collect the results
- Destroy the instance of the toolkit

Note that one instance of the toolkit may be used to scan multiple images on sequence, however in a multi-threaded environment each thread must use its own instance of the toolkit.

In the case when functions are being called directly from the DLL or shared object the code might look similar to:

```
hBarcode = mtCreateBarcodeInstance()

mtSetLicenseKey(hBarcode, "MY LICENSE KEY")

mtSetReadDataMatrix(hBarcode, true)

mtSetReadQRCode(hBarcode, true)

n = mtScanBarCode(hBarcode, imageFile)

or

n = mtScanBarCodeFromBitmap(hBarcode, bitmap)

(There are also functions for scanning from image files held in memory)


for (i = 1; i <= n; i++)
{
    value = mtGetBarString(hBarcode, i);
    type = mtGetBarStringType(hBarcode, i);
}

mtDestroyBarcodeInstance(hBarcode)
```

In the case when an object orientated interface is being used such as .Net or Java the code might look like:

```
object = new barcode object (syntax depends on interface)

object.LicenseKey = "MY LICENSE KEY"

object.ReadDataMatrix = true

object.ReadQRCode = true
```

```
n = object.ScanBarCode (imageFile)

or

n = object.ScanBarCodeFromBitmap(…)

There are also functions for scanning from image files held in memory

for (i = 1; i <= n; i++)
{
    value = object.GetBarString(hBarcode, i);
    type = object.GetBarStringType(hBarcode, i);
}

delete barcode object (syntax depends on interface)
```

4.1 .NET

The following options are available for .NET developers:

- .NET Standard 2.0 Nuget package (Windows, Linux, OSX)
- .NET Framework 4.0 Nuget package (Windows only)
- Reference the .NET Framework 4.0 Component DLL (Windows only and as used in the .NET Framework 4.0 Nuget package above)

The following table explains the compatibility of the above options…

	.NET Framework 4.0 - 4.6	.NET Framework >= 4.6.1	.NET Standard >= 2.0	.NET Core >= 2.0	.NET >= 5.0	Xamarin	Linux/OSX Compatible
.NET Standard Nuget package	N	Y	Y	Y	Y	Y	Y
.NET Framework Nuget package	Y	Y	N	N	N	N	N
.NET Framework Component DLL	Y	Y	N	N	N	N	N

4.1.1 .NET Standard Nuget package (Target = .NET Standard 2.0)

The NuGet package is called “softekbarcodenetstandard” and can be found at the following URL:

https://www.nuget.org/packages/softekbarcodenetstandard/

This is suitable for use with:

- .NET Framework (>= 4.6.1),

- .NET Core (>= 2.0)
- .NET Standard (>= 2.0)
- .NET (>= 5.0)
- .NET Xamarin

…and will run on Windows, OSX and Linux platforms. Android and iOS are not currently supported.

Add the package to your project and then use it with code similar to:

[csharp]

```
SoftekBarcodeNetStandard.BarcodeReader barcode = new SoftekBarcodeNetStandard.BarcodeReader();
barcode.LicenseKey = "set your license key here";
barcode.ReadDataMatrix = false;
barcode.ReadQRCode = true;
barcode.ReadCode128 = true;
int n = barcode.ScanBarCode("image.pdf");
for (int i = 1; i <= n; i++)
{
    Console.WriteLine("Value: " + barcode.GetBarString(i).ToString());
    Console.WriteLine("Type: " + barcode.GetBarStringType(i).ToString());
    Console.WriteLine("Page: " + barcode.GetBarStringPage(i).ToString());
}
```

[vb]

```
Dim barcode = New SoftekBarcodeNetStandard.BarcodeReader()
barcode.LicenseKey = "set your license key here"
barcode.ReadDataMatrix = False
barcode.ReadQRCode = True
barcode.ReadCode128 = True
Dim n = barcode.ScanBarCode("image.pdf")
Dim i As Integer
For i = 1 To n
    Console.WriteLine("Value: " + barcode.GetBarString(i).ToString())
    Console.WriteLine("Type: " + barcode.GetBarStringType(i).ToString())
    Console.WriteLine("Page: " + barcode.GetBarStringPage(i).ToString())
Next
```

Note that the .NET Standard component does not support reading directly from System.Drawing.Bitmap. If necessary, this can be achieved indirectly as follows:

```
System.Drawing.Bitmap bm = new System.Drawing.Bitmap(…);
System.Drawing.Imaging.BitmapData bmData = bm.LockBits(new System.Drawing.Rectangle(0, 0,
bm.Width, bm.Height),  System.Drawing.Imaging.ImageLockMode.ReadOnly,
System.Drawing.Imaging.PixelFormat.Format24bppRgb);
n = barcode.ScanBarCodeFromBitmap(bmData.Width, bmData.Height, bmData.Stride, 24,
bmData.Scan0);
```

4.1.2 .NET Framework Nuget Package

The NuGet package is called “SoftekBarcodeNet” and can be found at the following URL:

https://www.nuget.org/packages/SoftekBarcodeNet

This is suitable for use with .NET Framework (>= 4.0) on Windows only.

Add the package to your project and then use it with code similar to:

[csharp]

```
SoftekBarcodeNet.BarcodeReader barcode = new SoftekBarcodeNet.BarcodeReader();
barcode.LicenseKey = "set your license key here";
barcode.ReadDataMatrix = false;
barcode.ReadQRCode = true;
barcode.ReadCode128 = true;
int n = barcode.ScanBarCode("image.pdf");
for (int i = 1; i <= n; i++)
{
    Console.WriteLine("Value: " + barcode.GetBarString(i).ToString());
    Console.WriteLine("Type: " + barcode.GetBarStringType(i).ToString());
    Console.WriteLine("Page: " + barcode.GetBarStringPage(i).ToString());
}
```

[vb]

```
Dim barcode = New SoftekBarcodeNet.BarcodeReader()
barcode.LicenseKey = "set your license key here"
barcode.ReadDataMatrix = False
barcode.ReadQRCode = True
barcode.ReadCode128 = True
Dim n = barcode.ScanBarCode("image.pdf")
Dim i As Integer
For i = 1 To n
    Console.WriteLine("Value: " + barcode.GetBarString(i).ToString())
    Console.WriteLine("Type: " + barcode.GetBarStringType(i).ToString())
    Console.WriteLine("Page: " + barcode.GetBarStringPage(i).ToString())
Next
```

4.1.3 .NET Framework Component DLL

This is suitable for use with the .NET Framework (>= 4.0) on Windows only.

Add a reference to the SoftekBarcodeNet.dll file from the dotnet folder.

Note that when using the SoftekBarcodeNet.dll component the path to the native DLL files for the toolkit must be given when the object is created…

[csharp]

```
SoftekBarcodeNet.BarcodeReader barcode = new
SoftekBarcodeNet.BarcodeReader(@"\path\to\softek\barcode\toolkit\DLL");
barcode.LicenseKey = "set your license key here";
barcode.ReadDataMatrix = false;
barcode.ReadQRCode = true;
barcode.ReadCode128 = true;
int n = barcode.ScanBarCode("image.pdf");
for (int i = 1; i <= n; i++)
{
    Console.WriteLine("Value: " + barcode.GetBarString(i).ToString());
    Console.WriteLine("Type: " + barcode.GetBarStringType(i).ToString());
    Console.WriteLine("Page: " + barcode.GetBarStringPage(i).ToString());
}
```

[vb]

```
Dim barcode = New SoftekBarcodeNet.BarcodeReader("\path\to\softek\barcode\toolkit\DLL")
barcode.LicenseKey = "set your license key here"
barcode.ReadDataMatrix = False
barcode.ReadQRCode = True
barcode.ReadCode128 = True
Dim n = barcode.ScanBarCode("image.pdf")
Dim i As Integer
For i = 1 To n
    Console.WriteLine("Value: " + barcode.GetBarString(i).ToString())
    Console.WriteLine("Type: " + barcode.GetBarStringType(i).ToString())
    Console.WriteLine("Page: " + barcode.GetBarStringPage(i).ToString())
Next
```

4.2 Visual Studio C++ (Windows SDK)

The SoftekBarcodeDLL.h header file can be found in the include folder and the SoftekBarcodeDLL.lib and SoftekBarcode64DLL.lib files can be found in the lib folder. Sample code can be found in the cpp folder.

4.3 Delphi (Windows SDK)

A sample Delphi project can be found in the Delphi folder.

4.4 Java

The Java interface for the Softek Barcode Reader Toolkit is implemented through a wrapper class called Softek/Barcode which can be found under the java folder. Sample java code for using the class can be found in java/sample.java.

On Linux the JAVA_LIBRARY_PATH environment variable must include the folder containing the libbardecode.so or libbardecode.dylib file. The go.sh shell script shows how the sample can be run.

On Windows the PATH environment variable must include the folder containing the SoftekBarcodeDLL.dll or SoftekBarcode64DLL.dll file. The go_x86.bat and go_x64.bat files show how the sample can be run.

4.5 COM (Windows SDK)

The COM interface is implemented through the SoftekBarcodeCOM.dll and SoftekBarcode64COM.dll files which wrap around the SoftekBarcodeDLL.dll and SoftekBarcode64DLL.dll files. These files can be found the DLL folder.

The COM interface can be registered using regsvr32 as admin (see REGISTER_COM.BAT and REGISTER_COM64.BAT in the DLL folder) and code might typically look like:

```
Set barcode = Server.CreateObject("SoftekBarcodeCOM.Barcode")
barcode.ReadQRCode = True
barcode.ScanBarCode("input.jpg")
n = barcode.scanExitCode
If (nBarCode > 0) Then
    strBarcode = barcode.barstring(1)
End If
```

IMPORTANT: Note that the return value for ScanBarCode is obtained through the scanExitCode property.

4.6 Bardecode Shell tool (Linux andWindows)

The bardecode shell/DOS tool is a standard part of the toolkit on Linux and a separate product for Windows. On Windows the tool scans TIF, JPG and PDF documents and on Linux it scans TIF and JPG as standard with PDF scanning as a licensing option.

Usage:

bardecode -f image_file [options] [long options]

options are flags such as -m, which may or may not take an argument

long options are property name and value pairs, such as --ReadQRCode=1 (turn on QRCodes reading).

Options are:

-C max_length	Maximum length of barcode
-c min_length	Minimum length of barcode
-d scan_direction	Scan direction
-f image_file	Path to image file (with .tif or .jpg extension)
-g	Process files and folders from the XML file specified with -x flag. See below for more details.
-i page_index	Page number of image indexed from 1
-j jump	Frequency at which scan lines are sampled, default is 1
-L oversample_sep	Distance between the line samples when over-sampling is used. Default value is 3.
-M min_space_widt	Minimum allowed size in pixels for a space between bars
-m	Scan for multiple barcodes
-N noise_reduction	Advanced Noise Reduction (0 = off, typical value is 20). This is normal noise reduction with Despeckle.
-n noise_reduction	Noise Reduction level (0 = off, typical value is 20)
-O	Sample the scan lines in blocks of 3 with each line separated by oversample_sep lines.

-o min_occurs	Minimum number of matching scan lines, default 2
-P	Show page numbers
-p pref_occurs	Preferred number of matching scan lines, default 5
-q quiet_size	Number of pixels wide for the quiet zone around bar code
-R pattern	Barcodes must match regular expression defined by pattern
-r x1,y1,x2,y2,m	Rectangle for scanning
-S file_template	Split a multi-page TIF file into smaller TIF files.

-T color_threshold	Set the color threshold
-t barcode_types	Specifies the types of barcodes to read, barcode_types is one or more of "code39", "code128", "upca", "upce", "ean8", "code25", "ean13", "codabar", "qrcode", "pdf417", "datamatrix" and "databar" joined together with the | character.
-t any	Searches for a barcode of any type
-u upper_ratio	Defunct flag
-w	Print TIF Warnings to STDERR
-X xml_output	Output results to specified XML file
-x xml_input	Load settings from specified XML file

The long options give access to most other features of the toolkit and should be used as follows:

--PropertyName value

Use 1 or 0 for Boolean value properties.

e.g.

--MedianFilter 1

Examples:

Scan for a Code 39 barcode in a portrait orientation image:

bardecode -f file.tif -t code39 -d 5

Scan for a QRCode barcode:

bardecode -f file.tif -t qrcode

4.7 Python

A sample python script called sample.py can be found in the python folder. This script calls functions directly from the barcode toolkit DLL or shared object file.

Use CDLL to load the the shared object files on Linux, OSX and Windows 64 bit or WinDLL on 32 bit Windows. Please refer to the Python folder for sample code and note the following:

mtCreateBarcodeInstance returns a HANDLE to a barcode instance and the result type should be set as follows:

```
a.mtCreateBarcodeInstance.restype = ctypes.c_void_p
hBarcode = c_void_p(a.mtCreateBarcodeInstance())
```

Strings passed into functions need to be encoded as utf-8 similar to:

```
a.mtSetLicenseKey(hBarcode, 'YOUR LICENSE KEY'.encode("utf-8"))
nBarcodes = a.mtScanBarCode(hBarcode, inputFile.encode("utf-8"))
```

Functions that return strings should be used as per the following example:

```
a.mtGetBarString.restype = ctypes.c_char_p
barcodeValue = a.mtGetBarString(hBarcode, i)
print ("Value = ", barcodeValue.decode("utf-8"))
```

4.8 Perl

A Perl module for the toolkit is available on Linux and OSX. For further information please refer to the README file in the perl folder in the Linux/OSX download.

5 Evaluation

Temporary license keys can be obtained from sales@bardecode.com – without a key the SDK will either display a pop-up box each time it scans an image for a barcode or return barcode values with the last 3 characters replaced by * symbols. Should you require any assistance with the toolkit then please contact support@bardecode.com. Please refer to the section on licensing for information on how to apply a license key.

6 Licensing

6.1 Using a License Key

License key are applied either by setting the LicenseKey property before the ScanBarCode method is called or as an additional parameter to the bardecode shell tool.

For example:

barcode.SetLicenseKey(“MY LICENSE KEY”)
barcode.ScanBarCode(“input.tif”)…

or

mtSetLicenseKey(hBarcode, “MY LICENSE KEY”)
mtScanBarCode(“input.tif”)…
…

6.2 Types of License

The following table shows the various licenses available for the toolkit and the associated features:

	.Net Fram' Win (a)	.Net Std Win(b)	.Net Std Linux(c)	Win DLL (d)	Linux lib (e)	Java & Python (f)	Server (g)	TIF JPG (h)	PDF (i)	Shell tool (j)
Win SDK Desktop (1)	Y	Y	N	Y	N	Y	N	Y	N	N
Win SDK Server (2)	Y	Y	N	Y	N	Y	Y	Y	N	N
Win SDK Dist (3)	Y	Y	N	Y	N	Y	Y	Y	N	N
Win SDK Desktop PDF (4)	Y	Y	N	Y	N	Y	N	Y	Y	N
Win SDK Server PDF (5)	Y	Y	N	Y	N	Y	Y	Y	Y	N
Win SDK Dist PDF (6)	Y	Y	N	Y	N	Y	Y	Y	Y	N
Linux SDK Desktop (7)	N	N	Y	N	Y	Y	N	Y	N	Y
Linux SDK Server (8)	N	N	Y	N	Y	Y	Y	Y	N	Y
Linux SDK Dist (9)	N	N	Y	N	Y	Y	Y	Y	N	Y
Linux SDK Desktop PDF (10)	N	N	Y	N	Y	Y	N	Y	Y	Y
Linux SDK Server PDF (11)	N	N	Y	N	Y	Y	Y	Y	Y	Y
Linux SDK Dist PDF (12)	N	N	Y	N	Y	Y	Y	Y	Y	Y
Multi-platform (13)	N	Y	Y	N	N	N	Y	Y	Y	N
Win DOS (14)	N	N	N	N	N	N	Y	Y	Y	Y

Licenses:

1. Softek Barcode Reader Toolkit for Windows Desktop Developer/Run-time License
2. Softek Barcode Reader Toolkit for Windows Server License
3. Softek Barcode Reader Toolkit for Windows Unlimited Distribution License
4. Softek Barcode Reader Toolkit for Windows Desktop Developer/Run-time License with PDF Extension
5. Softek Barcode Reader Toolkit for Windows Server License with PDF Extension
6. Softek Barcode Reader Toolkit for Windows Unlimited Distribution License with PDF Extension
7. Softek Barcode Reader Toolkit for Linux/OSX/Android/iOS etc Desktop Developer/Run-time License
8. Softek Barcode Reader Toolkit for Linux/OSX/Android/iOS Server License
9. Softek Barcode Reader Toolkit for Linux/OSX/Android/iOS Unlimited Distribution License
10. Softek Barcode Reader Toolkit for Linux/OSX/Android/iOS Desktop Developer/Run-time License with PDF Extension
11. Softek Barcode Reader Toolkit for Linux/OSX/Android/iOS Server License with PDF Extension
12. Softek Barcode Reader Toolkit for Linux/OSX/Android/iOS Unlimited Distribution License with PDF Extension

13. Softek Multi-platform Barcode Reader Toolkit for .Net Standard License (includes PDF Extension)
14. Softek DOS Command Prompt Tool License

Features:

a) .Net Framework >= 4.0 on Windows platforms
b) .Net Standard 2.0 and compatible versions of .Net on Windows platforms
c) .Net Standard 2.0 and compatible versions of .Net on Linux and OSX
d) Access to the Windows DLL functions and wrappers for COM, Java, Python etc
e) Access to the Linux/OSX… static and shared object library functions and wrappers for Java, Perl, Python etc
f) Java and Python interfaces
g) Licensed for server versions of Windows or Linux
h) Support for TIF and JPG image files
i) Support for Adobe PDF files
j) Command prompt interface to the toolkit

7 PDF Extension

The PDF Extension allows the toolkit to read barcodes from PDF files and uses the Pdfium library (Copyright 2014 The PDFium Authors) to load documents as bitmaps. The interface for reading from PDF files is the same as for TIF and JPG files.

Note that use of the PDF Extension depends on having the correct type of license.

7.1 Properties used by the PDF Extension

The following properties are used by the PDF Extension:

- PdfiumPath – the path to the Pdfium DLL or shared object file. Note that this is optional (see below).
- PdfImageOnly – if true and a page only consists of a single image then the toolkit will attempt to extract the image directly rather than render the page.
- PdfDpi – the resolution of a rendered page.
- PdfBpp – the color depth of a rendered page.
- PdfPassword – the password for a PDF file

The following properties are not currently used in this version of the SDK:

- PdfImageExtractOptions
- PdfImageRasterOptions
- PdfLocking

7.2 NuGet package

The NuGet packages include all the Pdfium files necessary and require no configuration in order to use the PDF Extension.

7.3 Windows DLL

The Windows DLL interface will search for the Pdfium DLL file in the following sequence:

1. The DLL file specified by the PdfiumPath property of the toolkit (if used)
2. Current folder
3. A folder called DLL within the current folder
4. PATH environment variable

With the exception of (1) above:

For 32-bit systems the Pdfium DLL should be called either softekpdfium.dll or pdfium.dll and will be searched for in this order.

For 64-bit systems the Pdfium DLL should be called either softekpdfium64.dll or pdfium64.dll and will be searched for in this order.

7.4 Linux and OSX Lib

The Linux and OSX lib interfaces will search for the Pdfium shared object file in the following sequence:

1. The shared object file specified by the PdfiumPath property of the toolkit (if used)
2. Current folder
3. A folder called lib within the current folder
4. A folder called lib in the parent folder of the current folder (i.e. “../lib”)
5. LD_LIBRARY_PATH environment variable

With the exception of (1) above:

For Linux systems the Pdfium shared object file should be called either libsoftekpdfium.so or libpdfium.so and will be searched for in this order.

For OSX systems the Pdfium shared object file should be called either libsoftekpdfium.dylib or libpdfium.dylib and will be searched for in this order.

8 Contact Information

8.1 Sales

Softek Software's online store can be found at the following URL:

https://www.bardecode.com/en1/purchas-select-product-group/

For all sales enquiries please contact sales@bardecode.com or call +44 845 056 8856. Softek Software can provide quotations on request and can also accept payment by wire transfer or check.

8.2 Support

Softek Software offers free pre-sales support for the Barcode Reader Toolkit and all licenses include 12 months support and upgrade cover. Support is available by email during UK office hours with response times normally within 1 UK working day.

Softek Software online knowledge base can be found at the following URL:

https://www.bardecode.com/en1/category/knowledge-base/

For all support enquiries please contact support@bardecode.com

9 Supported Image Formats

9.1 TIFF

The following compressions are supported in the DLL, .Net (SoftekBarcodeNet.dll), COM, OCX and Java interfaces:

- Uncompressed
- LZW (Lempel-Ziv-Welch)
- Packbits (run length encoding)
- Jpeg
- ZIP
- CCITT Fax 3/Group 3
- CCITT Fax 4/Group 4

9.2 BMP (Windows only)

The SDK supports single plane BMP files in either 1-bit (black and white), 8-bit (gray scale), 24 bit (color) or 32-bit (color) format.

9.3 Adobe PDF

Please see the section on the PDF Extesnsion

9.4 Other formats

The SDK also support images in Jpeg (Windows and Linux), GIF (Winows only) and PNG (Windows only) format.

10 Supported Barcode Formats

10.1 1-D Barcode Formats

The following 1-D barcode formats are support by the SDK (with corresponding properties given in brackets):

- Codabar also known as Code 2 of 7, Codeabar, Ames Code, NW-7 and Monarch (ReadCodabar)
- Code 128 Symbol Sets A, B and C (ReadCode128)
- Code 128 Short Format (ReadShortCode128)
- Code 2 of 5 Datalogic (ReadCode25ni)
- Code 2 of 5 Iata1 (ReadCode25ni)
- Code 2 of 5 Iata2 (ReadCode25ni)
- Code 2 of 5 Industrial (ReadCode25ni)
- Code 2 of 5 Interleaved (ReadCode25)
- Code 2 of 5 Matrix (ReadCode25ni)
- Code 3 of 9 (ReadCode39)
- Code 3 of 9 Extended (ReadCode39 and ExtendedCode39)
- Code 93 (ReadCode93)
- EAN-8, European Article Number/International Article Number (ReadEAN8)
- EAN-13 and UPC-A, European Article Number/International Article Number (ReadEAN13)
- GS1-128, UCC-128, EAN-128 (ReadCode128)
- GS1-Databar (please see 2-D section below)
- Patch Code Symbols (ReadPatchCodes)
- UPC-A, Universal Product Code (ReadEAN13 and ReadUPCA)
- UPC-E, Universal Product Code (ReadUPCE)

10.2 2-D and Stacked Barcode Formats

The following 2-D and stacked barcode formats are also supported:

- QR-Code Model 1 and 2 (ReadQRCode)
- Data Matrix ECC200 sizes 8x8 to 144x144 (ReadDataMatrix)
- GS1-Databar or Reduced Space Symbology. Omnidirectional, Stacked Omnidirectional, Expanded, Expanded Stacked and Limited (ReadDatabar)
- Micro-PDF-417 (ReadMicroPDF417)
- PDF-417, Portable Data File (ReadPDF417)

IMPORTANT: DataMatrix ECC 000-140 is NOT SUPPORTED.

11 Performance Considerations

11.1 Single or Multiple Barcodes

There are 2 properties that control the number of barcodes the toolkit expects to find in an image file:

MultipleRead controls whether the toolkit is searching for a single barcode or multiple barcodes.

MaxBarcodesPerPage sets the maximum number of barcodes expected on a single page when MultipleRead is TRUE.

The order of speed is as follows:

1. Fastest: MultipleRead = FALSE (the default). The toolkit stops searching when the first barcode has been found.
2. Fast: MultipleRead = TRUE and MaxBarcodesPerPage is non-zero. The toolkit will stop processing a page as soon as the maximum number of barcodes has been found.
3. Slower: MultipleRead = TRUE and MaxBarcodesPerPage = 0. The toolkit searches every page of an image until all possible barcodes have been found.

11.2 Barcode Types

Any barcode types that are not required for a particular application should be disabled. This will reduce the probability of a false positive result and increase processing speed. 2-D barcodes such as PDF-417, QR-Code and Datamatrix take significantly more processing time than 1-D barcodes and should not be enabled if not required.

11.3 Color Images

Black and white images are faster to process than color images because the widths of the bars and spaces are clearly defined. The ColorProcessingLevel property controls how much time the toolkit puts in to handling a color image and ranges from 0 to 5 with a default of 2. Smaller values are faster.

11.4 Quiet Zone

The space around a barcode is known as the quiet zone and is important in efficient barcode recognition. The size of the quiet zone is defined by the QuietZoneSize property. The default of 0 calculates the quiet zone from the resolution of the image where as non-zero values define a specific number of pixels. Smaller values for the quiet zone will be slower than larger values.

12 Character Encoding

Character encoding is only generally an issue when calling functions directly from the DLL or shared object library because they do not return Unicode strings. Other interfaces such as .Net and COM handle this seamlessly so long as the Encoding property is not altered from its default value.

The toolkit represents barcodes values internally using UTF-8 encoding and this is the default format that will be returned by the DLL and shared object functions. UTF-8 encodes multi-byte characters in a byte array, using between 1 and 4 bytes per character depending on the Unicode value. This means that it is possible to encode everything from A-Z to Kanji characters in the same string.

Most barcode values can be encoded using 1 byte per character and therefore need no special decoding, however some 2-D barcodes can encode characters that require 2 or more bytes and need special decoding.

The following VB.Net function will convert the IntPtr value returned by the DLL function mtGetBarString (encoded using UTF-8) into a String:

```
Public Function ConvertUTF8IntPtrtoString(ByVal ptr As System.IntPtr) As String
        Dim l As Integer
        l = System.Runtime.InteropServices.Marshal.PtrToStringAnsi(ptr).Length
        Dim utf8data(l) As Byte
        System.Runtime.InteropServices.Marshal.Copy(ptr, utf8data, 0, l)
        Return System.Text.Encoding.UTF8.GetString(utf8data)
End Function
```

And the following function will do the same in VC++ (using MFC):

```
int CSampleBarcodeReaderDlg::ConvertUTF8Value(LPCSTR in, CString &out)
{
        int l = MultiByteToWideChar(CP_UTF8, 0, in, -1, NULL, 0);
        wchar_t *str = new wchar_t[l];
        int r = MultiByteToWideChar(CP_UTF8, 0, in, -1, str, l);
        out = str;
        delete str ;
        return r ;
}
```

13 Understanding Confidence Levels for Barcode Reading

As the toolkit scans the images in a document, it assigns a score to each barcode-like pattern it finds. The score for a very clear barcode will typically be in the region 20 to 100, depending on the size of the barcode and the settings used within the sdk. The PrefOccurrence and MinOccurrence properties specify the scores for the toolkit to use when deciding which barcodes to report and which to ignore. The PrefOccurrence property is the preferred occurrence for a barcode and defaults to value of 5. Any barcode with a score greater than or equal to PrefOccurrence will be reported by the toolkit. Some documents contain difficult to read barcodes with very low scores - lower than PrefOccurrence. In this case (where all of the barcodes have scores lower than PrefOccurrence) the toolkit will report the barcode with the highest score - so long as this figure is greater than or equal to MinOccurrence (minimum occurrence).

Consider the case of a 3 page document with barcodes on each page:

Page 1: Barcode 0001 with score 12
Page 2: Barcode 0002 with score 4
Page 3: Barcode 0003 with score 7

With default settings except for MultipleRead = true, the toolkit will only report the barcodes on pages 1 and 3 because the barcode on page 2 has a score less then PrefOccurrence (5).

If we now set PageNo = 2 (so we only scan page 2) then the barcode "0002" is reported, because no other barcodes were found with a score >= PrefOccurrence (5) and the score for this barcode is >= MinOccurrence (2).

With PageNo back to 0 (scan all pages) and PrefOccurrence = 4 the toolkit will report all 3 barcodes because all the scores are now >= PrefOccurrence (4).

In conclusion, the purpose of MinOccurrence is to allow the capture of poor quality barcodes whilst minimising the number of false positive readings in documents containing good quality barcodes.

14 Reading Barcodes from Color Images

In general, it is easier to read a barcode from a higher resolution bi-tonal image than from a lower resolution color image. Color images often have low contrast levels between the black and white bars and can often be degraded by compressions such as jpeg. This can make it difficult for a barcode reader to determine the relative widths of the black bars and white spaces, which can make it especially hard to decode barcode types such as Code 128 or UPC/EAN.

There are 2 key properties involved when scanning a color image for a barcode - ColorThreshold and ColorProcessingLevel. ColorThreshold can be used to manually set a threshold at which a pixel value is considered black or white, but it is normally left at the default value of 0, which allows the toolkit to assess the threshold level for itself through the ColorProcessingLevel property. This can vary from 0 to 5, with higher values yielding best results but taking longer than lower values.

15 Splitting Documents According to Barcode Position

The TifSplit feature of the toolkit allows you to use barcodes as document separators in both TIF and PDF documents. The input file is scanned for barcodes and then split into a number of smaller documents.

There are 2 properties that control how the input file is split:

Setting TifSplitPath turns the feature on and controls where the new documents will be created.

The path can contain the tokens %d and %s:

- %d is replaced by a 1-based sequence number
- %s is replaced by the value of the barcode separator

e.g c:\tmp\Output%d.tif will create files Output1.tif, Output2.tif etc

The format of the output document is determined by the file extension used for the TifSplitPath property. TIF files may be split into either TIF or PDF format where as PDF documents may only be split into PDF format.

TifSplitMode controls how the input pages are copied to the output documents.

- Mode=0 creates output files that contain a barcode on page 1 (the first output file will always start with page 1 of the input file).
- Mode=1 creates output files that contain a barcode on the last page (the last output file will always end with the last page of the input file).
- Mode=2 creates output files that contain no barcodes. A new output document is started each time the software finds a barcode in the input file.

Example:

Suppose there is a 6 page TIF file with barcodes on pages 2 and 5. The barcode on page 2 has the value "AAAAAA" and the barcode on page 5 has the value "BBBBBB"

```
barcode.TifSplitPath = "C:\tmp\Output%s_%d.tif"
barcode.TifSplitMode = 0
barcode.ScanBarCode(InputPath)
```

...will create 3 output files. Output_1.tif will contain page 1, OutputAAAAAA_2.tif will contain pages 2, 3 and 4, and OutputBBBBBB_3.tif will contain pages 5 and 6.

```
barcode.TifSplitPath = "C:\tmp\Output%s_%d.tif"
barcode.TifSplitMode = 1
barcode.ScanBarCode(InputPath)
```

...will create 3 output files. OutputAAAAAA_1.tif will contain pages 1 and 2, OutputBBBBBB_2.tif will contain pages 3, 4 and 5, and Output_3.tif will contain page 6.

```
barcode.TifSplitPath = "C:\tmp\Output%s_%d.tif"
barcode.TifSplitMode = 2
barcode.ScanBarCode(InputPath)
```

...will create 3 output files. Output_1.tif will contain page 1, OutputAAAAAA_2.tif will contain pages 3 and 4, and OutputBBBBBB_3.tif will contain page 6.

16 Tips on Reading Barcodes

Some images may require non-default values for certain properties. This section describes some common issues with barcode images and suggests some possible solutions. It may be the case that no single set of properties is able to process all documents in a batch, if so then it's worth considering using xml settings.

16.1 Skewed Barcodes

The toolkit can read many skewed barcodes using default settings. Please refer to the DeskewMode property for more information on how the toolkit handles skewed images.

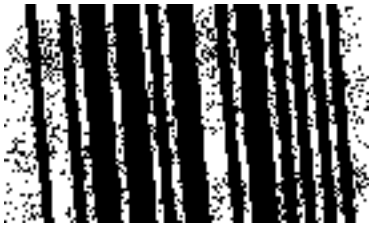
.

16.2 Badly defined edges to bars

In these cases the bars have joined together in certain places, which makes it difficult for the toolkit to separate the bars and determine their relative sizes.

Tip: Try setting NoiseReduction to a value between 10 and 20.

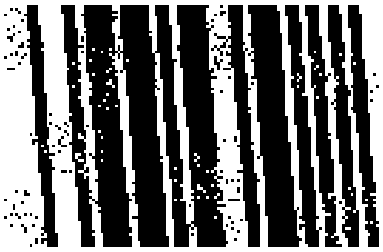
16.3 Noisy background to the image



Similar to the above problem except the entire image will typically contain black dots in the background.

Tip: Try setting NoiseReduction to a value between 5 and 10.

16.4 White speckles in the black bars



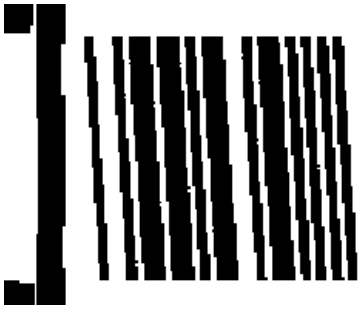
In these cases the image will typically contain black dots in the background and white dots in the black bars.

Tip: Try setting NoiseReduction to a value between 5 and 10 and Despeckle to True.

Black speckles in the spaces between bars

White speckles in the bars

Lines and other marks close to the left or right hand end of the barcode



Most barcodes should have a quite zone around the barcode, to distinguish the barcode from the rest of the image – however, as in the above example it's not unusual for barcodes to be printed in boxes which may result in the barcode being ignored by the toolkit.

Tip: Measure the distance in pixels between the barcode and the edge of the box and try setting QuietZoneSize to a value slightly less than this distance. Typically this might be a value between 10 and 20. Use values less than 5 with care because they can result in false positive readings for certain barcode types.

17 Reading Barcodes from Bitmaps Held in Memory

The toolkit can read barcodes from bitmps held in memory via a number of functions and methods all described in the manual page for ScanBarCodeFromBitmap.

Windows SDK:

mtScanBarCodeFromBitmap reads from a bitmap handle (HBITMAP)

mtScanBarCodeFromBitmapA reads from a pointer to the image data.

Linux/OSX:

mtScanBarCodeFromBitmap reads from a pointer to a BITMAP structure.

mtScanBarCodeFromBitmapA reads from a pointer to the image data.

mtScanBarCodeFromArray reads from a pointer to the image data.

.NET Framework:

ScanBarCodeFromBitmap reads from a System.Drawing.Bitmap object

ScanBarCodeFromBitmap reads from a byte array containing the image data

ScanBarCodeFromBitmap reads from an IntPtr pointing to the image data

.NET Standard:

ScanBarCodeFromBitmap reads from a byte array containing the image data

ScanBarCodeFromBitmap reads from an IntPtr pointing to the image data

Image data should be arranged so that the row stride is a multiple of 4 bytes.

Bits per pixel should be 1, 8, 24 or 32

18 Using XML Files to store sets of properties

It's possible that an application may need to try more than one set of properties on a set of images in order to maximize the read rate. One way to do this is to call ScanBarCode repeatedly, adjusting the settings between each call. Another way is to store the groups of settings in an xml file and call LoadXMLSettings prior to calling ScanBarCode. This causes ScanBarCode to load each page into a memory bitmap and then try each set of properties until a barcode is found – at which point the next page is loaded and the process repeated. The LoadXMLSettings reference page contains detailed information of the format of the xml file.

19 Appendix A: Methods Reference

In this section **Linux .so** refers to either the shared object or static versions of the library for Linux, OSX, Android, iOS and any other platform with a build available with the Linux download.

Note: All Windows DLL functions can be assumed to be declared with __stdcall

19.1 List of methods

CreateBarcodeInstance	create a instance of the barcode toolkit (win32 dll only)
DestroyBarcodeInstance	destroy an instance of the barcode toolkit (win32 dll only)
ExportXMLSettings	save settings to an xml file
GetBarCodeCount	get a count of bar codes found in a background scan
GetBarString	get a barcode value
GetBarStringBrx	get bottom right x coordinate of the bounding rectangle
GetBarStringBry	get bottom right y coordinate of the bounding rectangle
GetBarStringDirection	get the orientation of a barcode
GetBarStringQualityScore	get the quality score for a barcode
GetBarStringPage	get the page number for a barcode
GetBarStringTlx	get top left x coordinate of the bounding rectangle
GetBarStringTly	get top left y coordinate of the bounding rectangle
GetBarStringType	get the type of a barcode
GetLastError	get the last error number for the toolkit
GetLastWinError	get the last windows error number
GetProgress	get % progress of a background scan
GetScanExitCode	get the exit code of the last scan
LoadXMLSettings	load settings from an xml file
ProcessXML	process files and folders specified in an xml file
SaveResults	save the results of barcode reading to an xml or csv file
ScanBarCode	scan an image file for barcodes
ScanBarCodeFromBitmap	scan a bitmap held in memory for barcodes
ScanBarCodeFromByteArray	scan an image file held in memory for barcodes
ScanBarCodeFromDIB	scan a bitmap held in memory for barcodes
ScanBarCodeFromString	scan an image file held in memory for barcodes
ScanBarCodeAbort	abort a background scan
ScanBarCodeInbackground	scan an image file for barcodes in the background
ScanBarCodeWait	wait for a background scan
SetScanRect	specify the portion of an image to scan for barcodes

19.2 CreateBarcodeInstance

Syntax

Win dll: HANDLE **mtCreateBarcodeInstance**()
Linux .so: void ***mtCreateBarcodeInstance**()

Return Value

Handle to an instance of the toolkit.

Remarks

Create an instance of the barcode toolkit and return a handle that may be used with other functions. Note that in the object orientated interfaces to the toolkit this function is called automatically when the object is created.

See also:

DestroyBarcodeInstance

19.3 DestroyBarcodeInstance

Syntax

Win dll: int **mtDestroyBarcodeInstance**(HANDLE hBarcode)
Linux .so: int **mtDestroyBarcodeInstance**(void *hBarcode)

Parameters

hBarcode Handle to an instance of the barcode toolkit.

Return Value

0 on success and non-zero on failure.

Remarks

DestroyBarcodeInstance destroys an instance of the barcode toolkit and releases any resources used by the toolkit. Note that in the object orientated interfaces to the toolkit this function is called automatically when the object is deleted.

See also:

CreateBarcodeInstance

19.4 ExportXMLSettings

Syntax

.net/com/java/ocx: bool **ExportXMLSettings**(string filePath)
Win dll: int **mtExportXMLSettings**(HANDLE hBarcode, LPCSTR filePath)
Linux .so: int **mtExportXMLSettings**(void *hBarcode, char *filePath)

Parameters

hBarcode	Handle to an instance of the barcode toolkit
filePath	Path to the xml file to be created.

Return Value

1/True on success and 0/False on failure.

Remarks

ExportXMLSettings saves the current property values to an XML format file. The settings could be modified and loaded again using LoadXMLSettings.

The default set of properties is as follows:

```
<xml version='1.0' encoding='iso-8859-1'>
    <SoftekBarcode>
        <Properties>
            <AllowDuplicateValues>1</AllowDuplicateValues>
            <BarcodesAtTopOfPage>0</BarcodesAtTopOfPage>
            <BitmapResolution>200</BitmapResolution>
            <CodabarMaxVariance>20</CodabarMaxVariance>
            <Code128DebugMode>0</Code128DebugMode>
            <Code128Lenient>0</Code128Lenient>
            <Code128SearchLevel>2</Code128SearchLevel>
            <Code25Checksum>0</Code25Checksum>
            <Code25MinOccurrenceLength>7</Code25MinOccurrenceLength>
            <Code25PitchVariation>20</Code25PitchVariation>
            <Code39Checksum>0</Code39Checksum>
            <Code39MaxRatioPcnt>0</Code39MaxRatioPcnt>
```

<Code39NeedStartStop>1</Code39NeedStartStop>

<ColorChunks>1</ColorChunks>

<ColorProcessingLevel>2</ColorProcessingLevel>

<ColorThreshold>0</ColorThreshold>

<ConvertUPCEToEAN13>1</ConvertUPCEToEAN13>

<DataMatrixAutoUTF8>1</DataMatrixAutoUTF8>

<DataMatrixFinderGapTolerance>6</DataMatrixFinderGapTolerance>

<DataMatrixRectangleSupport>1</DataMatrixRectangleSupport>

<DataMatrixSearchLevel>3</DataMatrixSearchLevel>

<DeskewMode>2</DeskewMode>

<DeskewAfterNormalScan>0</DeskewAfterNormalScan>

<Despeckle>0</Despeckle>

<DotDataMatrixSupport>0</DotDataMatrixSupport>

<EdgeThreshold>20</EdgeThreshold>

<Encoding>3</Encoding>

<ErrorCorrection>0</ErrorCorrection>

<ExtendedCode39>0</ExtendedCode39>

<FastScanLineJump>25</FastScanLineJump>

<FixedLengthCode25>0</FixedLengthCode25>

<GammaCorrection>100</GammaCorrection>

<LineJump>1</LineJump>

<MaxBarcodesPerPage>0</MaxBarcodesPerPage>

<MaxLength>9999</MaxLength>

<MaxThreads>4</MaxThreads>

<MedianFilter>0</MedianFilter>

<MedianFilterLevel>1</MedianFilterLevel>

<MedianFilterBias>5</MedianFilterBias>

<MinSeparation>180</MinSeparation>

<MinLength>4</MinLength>

<Min2DLength>1</Min2DLength>

<MinOccurrence>2</MinOccurrence>

<MinResyncs>3</MinResyncs>

<MinSpaceBarWidth>0</MinSpaceBarWidth>

<MinThresholdDiff>30</MinThresholdDiff>

<MultipleRead>1</MultipleRead>

<NegativeImage>0</NegativeImage>

<NoiseReduction>0</NoiseReduction>

<PageNo>0</PageNo>

<PatchCodeMinOccurrence>30</PatchCodeMinOccurrence>

<Pattern></Pattern>

<Pdf417AutoUTF8>1</Pdf417AutoUTF8>

<Pdf417Debug>0</Pdf417Debug>

<Pdf417ChannelMode>0</Pdf417ChannelMode>

<PDF417MacroEscapeBackslash>1</PDF417MacroEscapeBackslash>

<PdfBpp>8</PdfBpp>

<PdfDpi>300</PdfDpi>

<PdfiumPath></PdfiumPath>

<PdfLocking>2</PdfLocking>

<PdfImageExtractOptions>15</PdfImageExtractOptions>

<PdfImageOnly>1</PdfImageOnly>

<PdfImageRasterOptions>0</PdfImageRasterOptions>

<PdfMaxMem>128</PdfMaxMem>

<Photometric>0</Photometric>

<UsePrePageLoadTimer>0</UsePrePageLoadTimer>

<PrefOccurrence>5</PrefOccurrence>

<QRCodeAutoUTF8>1</QRCodeAutoUTF8>

<QRCodeAutoMedianFilter>15</QRCodeAutoMedianFilter>

<QRCodeByteMode>2</QRCodeByteMode>

<QRCodeBWAutoMedianFilter>1</QRCodeBWAutoMedianFilter>

<QRCodeFinderTolerance>0</QRCodeFinderTolerance>

<QRCodeKanjiModeConvertUTF8>1</QRCodeKanjiModeConvertUTF8>

<QRCodeReadInverted>1</QRCodeReadInverted>

<QuietZoneSize>0</QuietZoneSize>

<QuotedPrintableCharSet>0</QuotedPrintableCharSet>

<ReadCodabar>1</ReadCodabar>

<ReadCode128>1</ReadCode128>

<ReadCode25ni>0</ReadCode25ni>

<ReadCode25>1</ReadCode25>

<ReadCode39>1</ReadCode39>

<ReadCode93>0</ReadCode93>

<ReadDatabar>0</ReadDatabar>

<ReadDataMatrix>0</ReadDataMatrix>

<ReadEAN8>1</ReadEAN8>

<ReadEAN13>1</ReadEAN13>

<ReadEAN13Supplemental>0</ReadEAN13Supplemental>

<ReadMicroPDF417>0</ReadMicroPDF417>

<ReadNumeric>0</ReadNumeric>

<ReadPatchCodes>0</ReadPatchCodes>

<ReadPDF417>0</ReadPDF417>

<ReadQRCode>0</ReadQRCode>

<ReadShortCode128>0</ReadShortCode128>

<ReadUPCA>0</ReadUPCA>

<ReadUPCE>1</ReadUPCE>

<ReportUnreadBarcodes>0</ReportUnreadBarcodes>

<RotateIfNoBarcode>0</RotateIfNoBarcode>

<SkewedLinear>1</SkewedLinear>

<ResyncChars>2</ResyncChars>

<ResyncRows>10</ResyncRows>

<ScanDirection>15</ScanDirection>

<ShortCode128MinLength>2</ShortCode128MinLength>

<SkewLineJump>9</SkewLineJump>

<SkewTolerance>0</SkewTolerance>

<ShowCheckDigit>0</ShowCheckDigit>

<ShowCodabarStartStop>1</ShowCodabarStartStop>

<SkewSpeed>3</SkewSpeed>

<TifSplitMode>0</TifSplitMode>

<TifSplitPath></TifSplitPath>

<TimeOut>5000</TimeOut>

<TwoDTimeOutPcnt>80</TwoDTimeOutPcnt>

<UseFastScan>1</UseFastScan>

<UseOld1DAlgorithms>1</UseOld1DAlgorithms>

<UseOldCode128Algorithm>0</UseOldCode128Algorithm>

<UseRunCache>1</UseRunCache>

<UseScaledFaxCoords>0</UseScaledFaxCoords>

<UseOverSampling>0</UseOverSampling>

<WeightLongerBarcodes>1</WeightLongerBarcodes>

</Properties>

</SoftekBarcode>

</xml>

19.5 GetBarCodeCount

Syntax

.net: int **GetBarCodeCount**()
com: int **BarCodeCount**
Win dll: int **mtGetBarCodeCount** (HANDLE hBarcode)
Linux .so: int **mtGetBarCodeCount** (void *hBarcode)

Not available in java and ocx interfaces.

Parameters

hBarcode Handle to an instance of the barcode toolkit.

Return Value

For a background scan this function returns the number of bar codes found by the scan. Once the scan completes the function GetScanExitCode should be used to check for errors and get the number of bar codes.

See also: ScanBarCodeInBackground, ScanBarCodeWait, ScanBarCodeAbort, GetProgress, GetScanExitCode

19.6 GetBarString

Syntax

.net:	string **GetBarString**(n)
	byte[] **GetRawBarStringBytes**(n)
java/ocx:	string **GetBarString**(n)
com:	string **BarString**(n)
Win dll:	LPCSTR **mtGetBarString** (HANDLE hBarcode, short n)
	int **mtGetBarStringA**(HANDLE hBarcode, short n, LPSTR ptr, int len)
	int **mtGetRawBarString**(HANDLE hBarcode, short n, LPSTR ptr, int len)
Linux .so:	unsigned char ***mtGetBarString**(void *hBarcode, int n)
	int **mtGetBarStringA**(void *hBarcode, int n, char *ptr, int len)
	int **mtGetRawBarString**(void *hBarcode, short n, unsigned char *ptr, int len)

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
n	1-based index to barcode to be queried.
ptr	buffer to receive barcode value
len	length of buffer

Return Value

GetBarString retrieves a barcode string that has been detected by the ScanBarCode method. In .net applications, use System.Runtime.InteropServices.Marshal.PtrToStringAnsi to convert the IntPtr returned by mtGetBarString to a String.

The functions mtGetBarStringA and mtGetRawBarString fill the memory pointed to by ptr with the contents of the barcode value and return the number of bytes copied or -1 on error (len should be the size of the memory pointed to by ptr). The function mtGetRawBarString may be used to obtain binary data that may include null characters. Note that the value of Encoding should NOT be set to 0 when using this function.

GetRawBarStringBytes returns an array of bytes in the .Net interface.

Remarks

- Check digit characters are only output if the ShowCheckDigit property is set to True.
- UPC-E barcodes are a zero-suppressed version of EAN-13 and as such are converted to EAN-13 format unless the ConvertUPCEToEAN13 property is set to False.
- Code 39 barcodes are not returned with the start/stop * characters.
- Codabar barcode value are always returned with the start/stop character pair, which can be either a/t, b/n, c/* or d/n.
- The following strings can be returned for patch codes - "Type I", "Type II", "Type III", "Type IV", "Type VI" and "Type T".

- When handling 2-D barcodes containing binary data it is worth disabling properties such as Pdf417AutoUTF8, QRCodeAutoUTF8 , QRCodeByteMode and DataMatrixAutoUTF8.

19.7 GetBarStringDirection

Syntax

.net/java/ocx: int **GetBarStringDirection**(int n)
com: int **BarStringDirection**(int n)
Win dll: short **mtGetBarStringDirection**(HANDLE hBarcode, short n)
Linux .so: int **mtGetBarStringDirection**(void *hBarcode, int n)

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
n	1-based index to barcode to be queried.

Return Value

1 = Left to Right

2 = Top to Bottom

4 = Right to Left

8 = Bottom to Top

16 = Top Left to Bottom Right

32 = Top Right to Bottom Left

64 = Bottom Right to Top Left

128 = Bottom Left to Top Right

Remarks:

Call GetBarStringDirection after calling ScanBarCode to determine the orientations of barcode found in a document. Note that this function will always return 1, 2, 4 or 8 for a QR-Code.

19.8 GetBarStringBrx and BarStringBottomRightX

Syntax

.net:	int **GetBarStringBrx**(int n)
com:	int **BarStringBottomRightX**(int n)
Win dll:	int **mtGetBarStringBrx**(HANDLE hBarcode, short n)
Linux .so:	int **mtGetBarStringBrx**(void *hBarcode, int n)

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
n	1-based index to barcode to be queried.

Return Value

The bottom right X coordinate of the bounding rectangle of the portion of the barcode read by the toolkit.

19.9 GetBarStringBry and BarStringBottomRightY

Syntax

.net:	int **GetBarStringBry**(int n)
com:	int **BarStringBottomRightY**(int n)
Win dll:	int **mtGetBarStringBry**(HANDLE hBarcode, short n)
Linux .so:	int **mtGetBarStringBry**(void *hBarcode, int n)

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
n	1-based index to barcode to be queried.

Return Value

The bottom right Y coordinate of the bounding rectangle of the portion of the barcode read by the toolkit.

19.10 GetBarStringPage and BarStringPage

Syntax

.net:	int **GetBarStringPage**(int n)
com:	int **BarStringPage**(int n)
Win dll:	int **mtGetBarStringPage**(HANDLE hBarcode, short n)
Linux .so:	int **mtGetBarStringPage**(void *hBarcode, int n)

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
n	1-based index to barcode to be queried.

Return Value

The page number in the input file where the barcode was located.

19.11 GetBarStringTlx and BarStringTopLeftX

.net: int **GetBarStringTlx**(int n)
com: int **BarStringTopLeftX**(int n)
Win dll: int **mtGetBarStringTlx**(HANDLE hBarcode, short n)
Linux .so: int **mtGetBarStringTlx**(void *hBarcode, int n)

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
n	1-based index to barcode to be queried.

Return Value

The top left X coordinate of the bounding rectangle of the portion of the barcode read by the toolkit.

19.12 GetBarStringTly and BarStringTopLeftY

.net: int **GetBarStringTly**(int n)
com: int **BarStringTopLeftY**(int n)
Win dll: int **mtGetBarStringTly**(HANDLE hBarcode, short n)
Linux .so: int **mtGetBarStringTly**(void *hBarcode, int n)

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
n	1-based index to barcode to be queried.

Return Value

The top left Y coordinate of the bounding rectangle of the portion of the barcode read by the toolkit.

19.13 GetBarStringQualityScore

Syntax

.net/java/ocx: int **GetBarStringQualityScore**(int n)
com: int **BarStringQualityScore**(int n)
Win dll: short **mtGetBarStringQualityScore**(HANDLE hBarcode, short n)
Linux .so: int **mtGetBarStringQualityScore**(void *hBarcode, int n)

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
n	1-based index to barcode to be queried.

Return Value

An integer between 1 and 5

Remarks:

Call GetBarStringQualityScore to get a score between 1 and 5 that indicates the quality of the barcode. A score of 1 indicates a low quality barcode and a score of 5 indicates a high quality barcode. The quality score depends on the clarity of the barcode in the image but can also vary as settings are changed within the toolkit, especially settings that add filters to the image prior to the recognition process. The quality score can be used to tune the recognition process both in terms of scanner settings and toolkit settings and can also be used in production to monitor for changes in the quality of the scanning process.

How to interpret the quality score:

For 1 –D barcodes:

1 = Very low quality barcode and similar barcodes may not decode. Take action to improve scan quality. Barcodes with a score of 1 may be omitted from results when other barcodes with a higher score are found in the same image.
2 = Low quality barcode. Take action to improve scan quality.
3 = Acceptable barcode quality but worth considering toolkit settings such as noise reduction, despeckle and median filter.
4 or 5 = Good quality barcode. No action required.

Common reasons for a low score with 1-D bar codes include:

- Black and/or white speckles in the bar code zone.
- Distortions in the width of the bars.
- Bars touching each other
- Broken bars – i.e. the spaces between bars touching each other.
- Skewed barcode.

A low score for a 2-D bar code will indicate that a high level of error correction was required. A score of 1 means recognition was at the borderline and any further reduction in quality would result in failure.

19.14 GetBarStringType

Syntax

.net/java/ocx: string **GetBarStringType**(int n)
com: string **BarStringType**(int n)
Win dll: LPCSTR **mtGetBarStringType**(HANDLE hBarcode, short n)
Linux .so: unsigned char ***mtGetBarStringType**(void *hBarcode, int n)

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
n	1-based index to barcode to be queried.

Return Value

GetBarStringType returns a string representing the type of barcode.

Remarks

Call GetBarStringType after calling ScanBarCode to determine the types of barcode found in a document. Examples of values returned by the function are “CODE39” and “CODE128”.

19.15 GetLastError

Syntax

.net/java/ocx: int **GetLastError()**
com: int **LastError()**
Win dll: int **mtGetLastError**()
Linux .so: int **mtGetLastError**()

Parameters

hBarcode Handle to an instance of the barcode toolkit (dll only).

Return Value

Please refer to the following table for an explanation of the possible error numbers:

```
1000  Failed to load SoftekBarcodePDF.dll
1001  Failed to find a function in SoftekBarcodePDF.dll
1002  Failed to get the path for the temp file folder
1003  Failed to generate a temporary name
1004  Failed to generate a temp file name
1005  Failed to load SoftekBarcodePDF.dll
1006  Failed to find a function in SoftekBarcodePDF.dll
1007  Failed to load SoftekBarcodePDF.dll
1008  Failed to find a function in SoftekBarcodePDF.dll
1009  Failed to open a BMP file
1010  Failed to read BMP file header
1011  Failed to read BMP file info
1012  Memory allocation problem when loading a bitmap
1013  Memory allocation problem when loading a bitmap
1014  Error creating bitmap handle
1015  Failed to create handle for URL
1016  Failed to open requested URL
1017  Failed to open temporary file for contents of URL
1018  File open failed.
1019  Null dib handle supplied to mtScanBarCodeFromDIB
1020  Failed to find a function in SoftekBarcodePDF.dll
1021  Failed to load SoftekBarcodePDF.dll
1024  Time out occurred for one of the pages processed
1025  Failed to create the Start Event for a thread
1026  Failed to create the End Event for a thread
1027  Failed to create a thread
1028  Failed to create a barcode instance for a thread
1029  Wait for event in thread failed
1030  Background scan is still active or hasn't been cleared with
1031  Background scan is still active or hasn't been cleared with
ScanBarCodeWait
1032  Background scan failed to start new thread
1033  Memory allocation error
1034  Warning for old style jpeg files
2000  Failed to load pdf2image.dll
2001  PDF conversion returned zero or negative page count
2002  Failed to load pdf2image.dll
2003  Failed to load a function in pdf2image.dll
2004  PDF conversion returned null handle
2005  Failed to load pdf2tif.dll
2006  Failed to find a function in pdf2image.dll
2007  Failed to find a function in pdf2image.dll
```

```
2008  Requested page number is out of range
2009  Requested page number is out of range
2010  PDF conversion failed
2011  PDf conversion failed
2012  Failed to find a function in pdf2image.dll
2013  Failed to find a function in pdf2image.dll
2014  Failed to load file as a PDF document - probably a format error
2015  Failed to generate full path name
2016  No access to document
2017  Cannot load PdftoImageGetPageBox function
2018  PDF file is encrypted
2019  Failed to load ghostscript library
2020  Failed to load ghostscript library function
2021  Conversion process failed
2100  Failed to load debenu library
2101  Failed to unlock debenu library
2102  Failed to create instance of debenu library
2103  Failed to load file into debenu library
2104  Failed to render a page to memory
2105  Failed to open TIF file
2106  Failed to render page to black and white
2107  Failed to load images from a page
2108  Failed to select page
2109  Failed to create bitmap from debenu data
2110  Failed to create bitmap from debenu data
2111  Failed to load Pdfium DLL from Debenu
2112  Failed to load Debenu
2113  Failed to load DebenuPDFRendererDLL
3000  Failed to connect to COM server PDF2ImageCOM.exe
3001  Failed to connect to COM server PDF2ImageCOM.exe
3002  Get page count function failed
3003  Get page box function failed
3004  Internal error in PDFToImageDrawToHDC
4000  Failed to connect to COM server PDF2TIFCOM.exe
4001  Failed to connect to COM server PDF2TIFCOM.exe
5000  Failed to load Pdfium dll or .so file
5001  Failed to load all necessary functions from the Pdfium dll or .so
file
5002  Failed to load the PDF document using Pdfium
5003  Failed to create the Pdfium bitmap
5004  Failed to load the PDF page using Pdfium
5005  Failed to create the mutex lock for access to Pdfium
5006  Failed to create the Pdfium image
5007  Failed to create the Pdfium page
5008  Pdfium SetBitmap function failed
5009  Failed to open the PDF document for splitting usiong Pdfium
5010  Start of split using Pdfium is out of range
5011  End of split using Pdfium is out of range
5012  Error creating destination split file using Pdfium
5013  Failed to import pages for a split using Pdfium
5014  Error when writing to destination split file using Pdfium
5015  Failed to write to TIF file when splitting PDF file with Pdfium
5016  License not valid for handling PDF file
```

Remarks:

Call GetLastError to retrieve the last internal error number for the toolkit .

19.16 GetLastWinError

Syntax

.net/java/ocx: int **GetLastWinError()**
com: int **LastWinError()**
Win dll: int **mtGetLastWinError**(HANDLE hBarcode)
(*) Linux .so: int **mtGetLastWinError**(void *hBarcode)

Parameters

hBarcode Handle to an instance of the barcode toolkit .

Return Value

The last windows error number, as recorded at the time of the last toolkit error. (*) Although this function exists on Linux it always returns 0.

Remarks:
Call GetLastWinError to retrieve the windows error number.

19.17 GetProgress

Syntax

.net:	**GetProgress()**
com:	**Progress**
Win dll:	int **mtGetProgress** (HANDLE hBarcode)
Linux.so:	int **mtGetProgress** (void *hBarcode)

Not available in java and ocx interfaces.

Parameters

hBarcode	Handle to an instance of the barcode toolkit.

Return Value

A value between 0 and 100 inclusive that indicates progress in a background scan.

Remarks:

For single page documents the value will either be 0 or 100.

19.18 GetScanExitCode

Syntax

.net: int **GetScanExitCode**()
com: **BackgroundExitCode**
Win dll: **mtGetScanExitCode** (HANDLE hBarcode)
Linux .so: **mtGetScanExitCode** (void *hBarcode)

Not available in java and ocx interfaces.

Parameters

hBarcode Handle to an instance of the barcode toolkit.

Return Value

Exit code from the last scan. Call GetScanExitCode after a background scan to obtain the number of bar codes or the error code. See ScanBarCode for more details on possible exit codes.

See also: ScanBarCodeInBackground, ScanBarCodeWait, ScanBarCodeAbort, GetProgress, GetBarCodeCount

19.19 LoadXMLSettings

Syntax

Loading from xml file:

.net/java: bool **LoadXMLSettings**(string file)
ocx: bool **LoadXMLSettings**(string file, bool silent)
com: **LoadXMLSettings**(string file, bool silent)
Win dll: int **mtLoadXMLSettings**(HANDLE hbarcode, LPCSTR file, unsigned char silent)
Linux .so: int **mtLoadXMLSettings**(void *hbarcode, unsigned char *file, unsigned char silent)

Loading from string:

.net/java: bool **LoadXMLSettings**(string xml)
com/ocx: int **LoadXMLSettings**(string xml, bool silent)
Win dll: int **mtLoadXMLSettings**(HANDLE hbarcode, LPCSTR xml, unsigned char silent)
Linux.so: int **mtLoadXMLSettings**(void *hbarcode, unsigned char *xml, unsigned char silent)

Parameters

hBarcode	Handle to an instance of the barcode toolkit (dll only).
file	Path to the xml file to be loaded.
xml	String holding the xml property values.
silent	Unsigned char or boolean value. 1/True will suppress error messages.

Return Value

1/True on success and 0/False on failure.

Remarks

Load settings from the specified XML file or string. This method can also be used to define sets of properties to be applied sequentially to an image until a barcode is located. It's also possible to specify properties that only apply to particular pages of an image. The silent parameter controls whether message message boxes will be displayed if there is a parsing error.

Note that when loading from a string the XML data can be in the abbreviated form:

"*<PropertyName>Value</PropertyName>*"

For the com object, use the XMLRetval property to obtain the return value for the function.

The simplest format for the xml is as follows:

```
<xml>
  <SoftekBarcode>
    <Properties>
```

```
      <PropertyName>PropertyValue</PropertyName>
    </Properties>
  </SoftekBarcode>
</xml>
```

Example:

Set MedianFilter to 1 (True) and set ReadCode25 to 0 (False).

```
<xml>
  <SoftekBarcode>
    <Properties>
      <MedianFilter>1</MedianFilter>
      <ReadCode25>0</ReadCode25>
    </Properties>
  </SoftekBarcode>
</xml>
```

The xml file can also be used to apply sets of properties sequentially to an image until a barcode is found:

This example first applies default values to an image. If no barcode is found then a MedianFilter is applied.

```
<xml>
  <SoftekBarcode>
    <Properties>
    </Properties>
    <Properties>
      <MedianFilter>1</MedianFilter>
    </Properties>
  </SoftekBarcode>
</xml>
```

This example restricts the search to the first 3 pages of an image, until a barcode is located:

```
<xml>
  <SoftekBarcode>
    <Properties>
      <PageNo>1</PageNo>
    </Properties>
    <Properties>
      <PageNo>2</PageNo>
    </Properties>
    <Properties>
      <PageNo>3</PageNo>
    </Properties>
```

```
  </SoftekBarcode>
</xml>
```

The default set of properties can be found in the manual page for ExportXMLSettings.

See Also

ExportXMSettings ProcessXML

19.20 ProcessXML

Syntax

.net: bool **ProcessXML** (string inputFile, string outputFile)
ocx: bool **ProcessXML** (string inputFile, string outputFile, bool silent)
com: **ProcessXML** (string inputFile, string outputFile, bool silent)
Win dll: int **mtProcessXML** (HANDLE hbarcode, LPCSTR inputFile, LPCSTR outputFile, unsigned char silent)
Linux .so: int **mtProcessXML** (void *hbarcode, unsigned char *inputFile, unsigned char *outputFile, unsigned char silent)

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
inputFile	Path to the xml file to be processed.
ouputFile	Path to the xml or csv file to be created.
silent	Integer or boolean value. 1/True will suppress error messages.

Return Value

1/True on success and 0/False on failure. For the COM object, use the XMLRetval property to obtain the return value for the function.

Remarks

ProcessXML takes property values, file names and folder names from the inputFile and creates outputFile in either XML or CSV file format.

The property values defined in the XML file follow the specification as defined in the manual page for LoadXMLSettings. The files and folder to process are defined in the following way....

To process a file:

```
<Process>
  <File>/path/to/image.tif</File>
</Process>
```

To process the images in a folder:

```
<Process>
  <Folder>/path/to/folder</Folder>
</Process>
```

Both the File and Folder tags accept an optional id attribute. This can be used in the output to identify the image file or folder.

Example:

```
<xml>
  <SoftekBarcode>
```

```
    <Properties>
    </Properties>
    <Properties>
      <MedianFilter>1</MedianFilter>
    </Properties>
    <Process>
      <File id=1001>C:\tmp\image1.tif</File>
      <File id=1002>C:\tmp\image2.tif</File>
      <File id=1003>C:\tmp\image3.tif</File>
      <Folder id=9001>C:\tmp\images</Folder>
    </Process>
  </SoftekBarcode>
</xml>
```

See the manual page for SaveResults for the output file format.

19.21 SaveResults

Overview

.net/ocx: bool **SaveResults** (string file)
com: **SaveXMLResults** (string file)
Win dll: int **mtSaveResults** (HANDLE hbarcode, LPCSTR file)
Linux .so: int **mtSaveResults** (void *hbarcode, unsigned char *file)

Parameters

hBarcode Handle to an instance of the barcode toolkit (dll only).
file Path to the xml or csv file to be created.

Return Value

1/True on success and 0/False on failure.

Remarks

SaveResults saves the current set of results to an XML or CSV file.

The outputFile format is determined by the file extension (csv or xml).

The CSV fields are as follows:

Folder - location of image file
File Name
Count - number of barcodes found in image
Index - Index for this bar code (1,2,3 etc or -1 is no barcode found).
Error - Error number
ID - as specified in an xml input file for image or folder (see ProcessXML)
Value - value of barcode
Type - type of barcode
Hits - hit count for this barcode
Page - page number in image
Direction - ScanDirection mask value for this barcode
TopLeftX - x position for top left
TopLeftY - y position for top left
BottomRightX - x position for bottom right
BottomRightY - y position for bottom right

The XML output format contains the same information as the CSV format, but arranged in the following way:

```
<xml version='1.0' encoding='iso-8859-1'>
  <SoftekBarcode>
    <Result>
      <Folder>location of image file</Folder>
      <FileName>name of file</FileName>
```

```
      <Count>number of barcodes</Count>
      <Error>error number</Error>
      <ID>id as specified in an xml input file for image or folder (see ProcessXML)</ID>
      Repeated for each barcode found in image
      <Barcode>
        <Value>value of barcode</Value>
        <Type>type of barcode</Type>
        <Hits>score</Hits>
        <Page>page number</Page>
        <Direction>ScanDirection mask value for this barcode</Direction>
        <TopLeftX>x position for top left</TopLeftX>
        <TopLeftY>y position for top left</TopLeftY>
        <BottomRightX>x position for bottom right</BottomRightX>
        <BottomRightY>y position for bottom right</BottomRightY>
      </Barcode>
    </Result>
  </SoftekBarcode>
</xml>
```

19.22 ScanBarCode

Syntax

.net/java/ocx:	int **ScanBarCode**(string file)
com:	**ScanBarCode**(string file)
Win dll:	int **mtScanBarCode**(HANDLE hBarcode, LPCSTR file)
Linux .so:	int **mtScanBarCode**(void *hBarcode, unsigned char * file)

Parameters

hBarcode	Handle to an instance of the barcode toolkit (dll only).
file	Path to the file containing the image to be scanned for barcodes.

Return Value

-1	Error opening file
-2	BMP file is multi-plane
-3	Invalid number of bits per sample
-4	Memory allocation error
-5	Invalid tif photometric property
-6,-7,-8	Invalid license key.

Remarks

Scan the specified image file for bar code strings and return the number of bar codes found in the file. Note that the function will stop when the first barcode is found in a document unless the MulipleRead property is set to True.

Further information for errors can be found by calling the GetLastError and GetLastWinError functions.

When using the COM object call GetBarCodeCount to retrieve the number of barcodes found in the image.

See also: ScanBarCodeInBackground

GetBarCodeCount

19.23 ScanBarCodeFromBitmap

Syntax

.net (both standard and framework)

int **ScanBarCodeFromBitmap**(int width, int height, int rowStride, int bitsPixel, IntPtr bits, int length)
int **ScanBarCodeFromBitmap**(int width, int height, int rowStride, int bitsPixel, byte[] bits)

.net framework

int **ScanBarCodeFromBitmap**(System.Drawing.Bitmap bitmap)

ocx

int **ScanBarCodeFromBitmap**(HBITMAP hBitmap)

com

ScanBarCodeFromBitmap(HBITMAP hBitmap)

Win dll

int **mtScanBarCodeFromBitmap**(HANDLE hBarcode, HBITMAP hBitmap)
short **mtScanBarCodeFromBitmapA** (HANDLE hBarcode, int bmType, int width, int height, int rowStride, int bmPlanes, int bitsPixel, void *bits)

Linux .so

int **mtScanBarCodeFromBitmap**(void *hBarcode, BITMAP *pBitmap)
int **mtScanBarCodeFromBitmapA**(void *hBarcode, int bmType, int width, int height, int rowStride, int bmPlanes, int bitsPixel, unsigned char *bits)
int **mtScanBarCodeFromArray**(void *hBarcode, unsigned char *bits, int width, int height, int bitsPixel)

Parameters

hBarcode	Handle to an instance of the barcode toolkit
bitmap	System.Drawing.Bitmap object
hBitmap	handle (HBITMAP) to a bitmap object
pBitmap	Pointer to a BITMAP structure (see below)
width	width of bitmap in pixels
height	height of bitmap in pixels
rowStride	Number of bytes per row (see important note below)
bitsPixel	Number of bits per pixel
bits	Pointer or byte array containing the bitmap image data
length	Length of bitmap data
bmType	Must be 0
bmPlanes	Must be 1

Return Value

-1	N/A
-2	Bitmap is multi-plane
-3	Invalid number of bits per sample
-4	Memory allocation error
-5	N/A
-6,-7,-8	Invalid license key.

Remarks

Scan the specified bitmap and return the number of barcodes found or an error.

IMPORTANT: With the exception of the .net interface the bitmap data should be arranged so each row in the bitmap falls on a 4 byte boundary. For example, a bitmap that is 90 pixels wide at 24 bits per pixel requires 2 padding bytes at the end to ensure the rows fall on a 4 byte boundary (90 * 24 / 8 = 270, 270 / 4 = 67.5, 4 * 68 = 272). If rowStride is not a multiple of 4 then it is increased to the next multiple. Beware of increasing the value of the bitmap image width to include padding bytes since this can effectively introduce a white or black line into the far right of the image. The Linux function mtScanBarCodeFromArray calculates the rowStride from the bitmap width and bitsPerPixel and then increases the value to be a multiple of 4.

The default photometric interpretation for a 1 bit per pixel bitmap is 1 = black, 0 = white. This can be changed by setting the Photometric property.

When using a bitmap handle in Windows: The image must be single plane. Note that hBitmap is a HANDLE to a BITMAP, not the address of a BITMAP structure. The Windows GDI function, CreateBitmapIndirect can be used to create an HBITMAP from a BITMAP structure.

The BITMAP structure where used is defined as follows:

```
typedef struct tagBITMAP {
  int     bmType;
  int     bmWidth;
  int     bmHeight;
  int     bmWidthBytes;
  unsigned char    bmPlanes;
  unsigned char    bmBitsPixel;
  void       *  bmBits;
} BITMAP;
```

See also: ScanBarcodeFromDIB

Photometric

19.24 ScanBarCodeFromDIB

Syntax

com/ocx:	int **ScanBarCodeFromDIB**(hDIB)
Win dll:	int **mtScanBarCodeFromDIB**(HANDLE hBarcode, HANDLE hDIB)

Not supported in .net or Linux

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
hDIB	Memory object – see Remarks below.

Return Value

-1	N/A
-2	DIB is multi-plane
-3	Invalid number of bits per sample
-4	Memory allocation error
-5	N/A
-6,-7,-8	Invalid license key.

Remarks

Scan the specified device independent bitmap for bar code strings and return the number of bar codes found. The image must be single plane. The hDIB parameter is a handle to a memory object which has the same format as a BMP file not including the BITMAPFILEHEADER. Note that the function will stop when the first barcode is found in a document unless the MulipleRead property is set to True.

See also ScanBarCodeFromBitmap

19.25 ScanBarCodeFromString, ScanBarcodeFromByteArray

Syntax

.net:	int **ScanBarCodeFromByteArray**(byte[] data)
com:	int **ScanBarCodeFromByteArray**(BYTE *data, ULONG sz)
Win dll:	int **mtScanBarCodeFromString**(HANDLE hBarcode, unsigned char *data, size_t sz)
Linux .so:	int **mtScanBarCodeFromString**(void *hBarcode, unsigned char *data, size_t sz)

Parameters

hBarcode	Handle to an instance of the barcode toolkit (dll only).
data	A pointer to the image data or a byte array for .Net
sz	The size in bytes of the data.

Return Value

-1	Error opening file
-2	BMP file is multi-plane
-3	Invalid number of bits per sample
-4	Memory allocation error
-5	Invalid tif photometric property
-6,-7,-8	Invalid license key.

Remarks

Scans an image file held in memory. The data can be in TIF, JPG or PDF (*) file format and should be identical to the format of such a file on disc. The function scans for bar code strings and return the number of bar codes found in the file.

To scan memory bitmaps please use the function ScanBarCodeFromBitmap.

Further information for errors can be found by calling the GetLastError and GetLastWinError functions.

See also: ScanBarCode, ScanBarCodeFromStringInBackground

19.26 ScanBarCodeFromStringInBackground, ScanBarcodeFromByteArrayInBackground

Syntax

Win dll: short **mtScanBarCodeFromStringInBackground**(HANDLE hBarcode, unsigned char *data, size_t sz)

Not available in the Linux .so, com, ocx, java or .net interfaces.

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
data	A pointer to the image data or a byte array for .Net.
sz	The size in bytes of the data.

Return Value

-1	Error starting background scan
0	Success

Remarks

Start a background scan of the specified image file for one or more bar codes. Note that it is not possible to run 2 background scans at the same time and a background scan is not considered to have completed until the ScanBarCodeWait function has returned 0. Related functions are:

ScanBarCodeWait - wait (with time out) for background scan to finish.

ScanBarCodeAbort – request cancel of background scan.

GetProgress – returns percentage progress through scan (currently only returns 0 or 100 for a single page document).

GetBarCodeCount – returns number of bar codes so far during a background scan.

GetScanExitCode – get the exit code of the last background scan

See also: ScanBarCodeInBackground

19.27 ScanBarCodeAbort

Syntax

.net(*)/com: int **ScanBarCodeAbort**()
Win dll: int **mtScanBarCodeAbort**(HANDLE hBarcode)

*Not available in the Linux .so (including .net on Linux), ocx and java interfaces.

Parameters

hBarcode	Handle to an instance of the barcode toolkit.

Return Value

-1	Error aborting scan or no background scan in progress to abort
0	Request to abort scan accepted.

Remarks

Request an abort of a background scan. Note that it is still necessary to call ScanBarCodeWait to end the background scan correctly.

See also: ScanBarCodeInBackground, ScanBarCodeWait, GetProgress, GetBarCodeCount, GetScanExitCode

19.28 ScanBarCodeInBackground

Syntax

.net(*)/com:	int **ScanBarCodeInBackground**(string file)
Win dll:	int **mtScanBarCodeInBackground**(HANDLE hBarcode, LPCSTR file)

*Not available in the Linux .so (including .net on Linux), ocx and java interfaces.

Parameters

hBarcode	Handle to an instance of the barcode toolkit (dll only).
file	Path to the file containing the image to be scanned for barcodes.

Return Value

-1	Error starting background scan
0	Success

Remarks

Start a background scan of the specified image file for one or more bar codes. Note that it is not possible to run 2 background scans at the same time and a background scan is not considered to have completed until the ScanBarCodeWait function has returned 0. Related functions are:

ScanBarCodeWait - wait (with time out) for background scan to finish.

ScanBarCodeAbort – request cancel of background scan.

GetProgress – returns percentage progress through scan (currently only returns 0 or 100 for a single page document).

GetBarCodeCount – returns number of bar codes so far during a background scan.

GetScanExitCode – get the exit code of the last background scan

Examples:

The following code starts a background scan, waits until it finishes and then gets the exit code. It is the same as calling the ScanBarCode function:

```
barcode.ScanBarCodeinBackground("somefile.tif")
barcode.ScanBarCodeWait(-1)
n = barcode.GetScanExitCode()
```

A background scan will normally be started and then monitored using a timer. The function to check on the status of the scan might look like:

```
if (barcode.ScanBarCodeWait(0) == 1)
{
        percent = barcode.GetProgress()
```

```
        count = barcode.GetBarCodeCount()
        return
}

exitCode = barcode.GetScanExitCode()
```

Note that it may be preferable to use a small timer interval and call ScanBarCodeWait(N) where N is the number of milliseconds to block for between updates of percent and count since this will detect the completion of a scan faster.

For example:

```
if (barcode.ScanBarCodeWait(100) == 1)
{
        percent = barcode.GetProgress()
        count = barcode.GetBarCodeCount()
        return
}

exitCode = barcode.GetScanExitCode()
```

Note that after calling ScanBarCodeAbort it is still necessary to wait until ScanBarCodeWait returns 0 or -1 to clear the way for the next background scan to start.

Further information for errors can be found by calling the GetLastError and GetLastWinError functions.

See also: ScanBarCodeWait, ScanBarCodeAbort, GetProgress, GetBarCodeCount, GetScanExitCode

19.29 ScanBarCodeWait

Syntax

.net(*)/com: int **ScanBarCodeWait**(int ms)
Win dll: **mtScanBarCodeWait** (HANDLE hBarcode, int ms)

*Not available in the Linux .so (including .net on Linux) ocx and java interfaces.

Parameters

hBarcode	Handle to an instance of the barcode toolkit.
ms	See remarks below.

Return Value

0	background scan has completed
-1	background scan is no longer active
1	background scan is still active

Remarks

ScanBarCodeWait is used to see if a background scan has completed. If ms is -1 then the function will block until the background scan completes. If ms is 0 then the function will check on the status of the scan and return immediately. If ms > 0 then the function will block for the specified number of milliseconds or until the background scan completes.

See also: ScanBarCodeInBackground, ScanBarCodeWait, ScanBarCodeAbort, GetProgress, GetBarCodeCount, GetScanExitCode

19.30 SetScanRect

Syntax

.net/com/java/ocx:	SetScanRect(int tlx, int tly, int brx, int bry, int mode)
Win dll:	bool mtSetScanRect(HANDLE hBarcode, long tlx, long tly, long brx, long bry, short mode)
Win dll:	bool mtSetScanRectA(HANDLE hBarcode, int tlx, int tly, int brx, int bry, short mode)
Linux .so:	int mtSetScanRect(void *hBarcode, long tlx, long tly, long brx, long bry, int mode)
Linux .so:	int mtSetScanRectA(void *hBarcode, int tlx, int tly, int brx, int bry, int mode)

Parameters

hBarcode	Handle to an instance of the barcode toolkit (dll only).
tlx	x coordinate of top left hand corner
tly	y coordinate of top left hand corner
brx	x coordinate of bottom right hand corner
bry	y coordinate of bottom right hand corner
mode	mapping mode (see below)

Return Value

1/True on success and 0/False on failure.

Remarks

SetScanRect specifies the bounding rectangle in the image that should be searched for barcodes. To clear the rectangle and search the entire image set the rectangle to (-1, -1, -1, -1). The top left hand corner of an image is (0,0).

The mapping mode can have the following values:

0 = All measurements are in pixels.

1 = All measurements are a percentage of the width or height of the image.

20 Appendix B: Properties Reference

(*) indicates an advanced parameter than can only be set using LoadXMLSettings

2DTimeOutPcnt	percentage of TimeOut when 2-D barcode processing stops
AllowDuplicateValues	allow duplicate barcode values on the same page
BarcodesAtTopOfPage	limited number of barcodes situated at the top of a pages
BitmapResolution	set the resolution of a bitmap
CodabarMaxVariance (*)	max width variance for codabar characters
Code25Checksum	handle final character of code 25 barcode as checksum
Code25PitchVariation(*)	maximum percentage difference in width of Code 25 characters
Code39Checksum	handle final character of code 39 barcode as checksum
Code39MaxRatioPcnt	Max ratio between Code 39 narrow and wide bars
Code39NeedStartStop	expect start/stop characters with code 39 barcodes
ColorChunks (*)	divide scan lines into sections for threshold levels
ColorProcessingLevel	control the time spent processing a color image
ColorThreshold	set the color threshold level for a color image
ConvertUPCEToEAN13	automatically convert UPC-E format to EAN-13
DatabarOptions	set options for GS1 Databar barcodes
DataMatrixAutoUTF8	automatically sense UTF-8 encoded datamatrix bar codes
DotDataMatrixSupport	support dot datamatrix with larger spaces
DataMatrixFinderGapTolerance	maximum size of a gap in the datamatrix finder pattern.
DataMatrixRectangleSupport	read extended data matrix formats.
DataMatrixSearchLevel	balance speed against read rate for data matrix bar codes
DeskewAfterNormalScan	do upright scan before considering any other angles
DeskewMode	how to handle skewed documents
Despeckle	remove speckled marks from an image before scanning
Encoding	set the encoding method for barcode values
ErrorCorrection	attempt to correct errors
ExtendedCode39	assume extended code 39 barcode format

FastScanLineJump	frequency of line sampling for fast scans
FilePathEncoding	specify how file paths are encoded
GammaCorrection	set gamma correction level for color images
LicenseKey	set the license key
LineJump	control the frequency of line sampling
MaxBarcodesPerPage	maximum number of barcodes expected on a single page
MaxLength	set the maximum length for a barcode
MaxThreads	maximum number of threads during a scan
MedianFilter	perform a median filter on the image before scanning
MedianFilterBias(*)	set the black/white balance for median filters on 2-color images
MinLength	set the minimum length for a barcode
MinOccurrence	specify the lowest permitted score for a barcode
MinSeparation	minimum distance between barcodes of same value
MinSpaceBarWidth	minimum size of a space between bars
MultipleRead	scan for more than one barcode
NoiseReduction	perform noise reduction before scanning
PageNo	set the page number to scan in a muli-page image
PatchCodeMinOccurrence	minimum score for a Patch Code barcode.
Pattern	only report barcodes that fit the specified pattern
Pdf417AutoUTF8	Automatically sense UTF-8 encoded Pdf417 bar codes.
Pdf417Debug (*)	enable debug mode for PDF-417 barcodes
Pdf417ChannelMode(*)	set the channel mode for PDF-417 barcodes
PDF417MacroEscapeBackslash(*)	escape back-slash when Pdf417ChannelMode is 1
PdfBpp	load pdf documents at specified bits-per-pixel
PdfDpi	load pdf documents at specified dots-per-inch
PdfImageExtractOptions	mask to control how images are extracted from pdf files
PdfImageOnly	PDF documents will only contain images

PdfImageRasterOptions	mask to control how pdf files are rasterized
PdfPassword	password for a PDF document
Photometric	set photometric interpretation for bi-tonal bitmaps
PrefOccurrence	specify the preferred score for a barcode
QRCodeAutoUTF8	automatically sense UTF-8 encoded QR-Codes
QRCodeBWAutoMedianFilter(*)	Use median filter on black and white QR-Code reading
QRCodeReadInverted	Read white on black QR-Codes.
QuietZoneSize	set the size of the quiet zone around a barcode
QuotedPrintableCharSet	character set for quoted printable encoding
ReadCodabar	scan for codabar barcodes
ReadCode128	scan for code-128 barcodes
ReadCode25	scan for code-25 barcodes
ReadCode25ni	scan for non-interleaved code-25 barcodes
ReadCode39	scan for code-39 barcodes
ReadCode93	scan for code-93 barcodes
ReadDatabar	scan for databar barcodes
ReadDataMatrix	scan for datamatrix barcodes
ReadEAN13	scan for ean-13 barcodes
ReadEAN13Supplemental	Read EAN-13 supplemental data
ReadEAN8	scan for ean-8 barcodes
ReadMicroPDF417	scan for micro pdf-417 barcodes
ReadNumeric	only read numeric barcodes
ReadPatchCodes	scan for patch codes
ReadPDF417	scan for pdf-417 barcodes
ReadQRCode	scan for QR-codes.
ReadShortCode128	scan for short code-128 barcodes
ReadUPCA	scan for upc-a barcodes

ReadUPCE	scan for upc-e barcodes
ReportUnreadBarcodes	report barcodes that could not be otherwise decoded
ScanDirection	specify the orientations in which to scan
ShortCode128MinLength	set the minimum length for a short code-128 barcode
ShowCodabarStartStop	include codabar start/stop characters
ShowCheckDigit	display check digits where possible
SkewedDataMatrix	read skewed datamatrix barcodes
SkewedLinear	read skewed linear barcodes
SkewLineJump	frequency of line sampling for skewed barcodes
SkewTolerance	maximum tolerance for skewed barcodes
TifSplitMode	control the way that tif and pdf documents are split
TifSplitPath	specify the output path for splitting tif and pdf documents
TimeOut	time out for processing a single page
UseFastScan	perform an initial fast scan of an image
UseOldCode128Algorithm (*)	use the old code-128 detection method
UseOverSampling	process multiple scan lines at the same time
UseRunCache (*)	use a memory cache for run-length information
WeightLongerBarcodes	accept lower scores for longer barcodes

(*) indicates an advanced parameter than can only be set using LoadXMLSettings.

20.1 Setting and Getting Property Values

All properties, except those marked as “Advanced” can be set and read using the following methods. All properties (including those marked as “Advanced”) can also be set using LoadXMLSettings.

Setting a property value using the .net/com or java interfaces:

Object.*PropertyName* = Value

Getting a property value using the .net/com or java interfaces:

Value = Object.*PropertyName*

Setting a property value using the ocx interface:

Object.Set*PropertyName*(Value)

Getting a property value using the ocx interface:

Value = Object.Get*PropertyName*()

Setting a property value using the dll interface:

mtSet*PropertyName*(hBarcode, Value)

Getting a property value using the dll interface:

Value = mtGet*PropertyName*(hBarcode)

Win32 dll declarations:

If the property is of type BOOL:

VB.Net:

Private Declare Function mtGet*PropertyName* Lib “SoftekBarcodeDLL” (ByVal hBarcode As System.IntPtr) As Boolean
Private Declare Function mtSet*PropertyName* Lib “SoftekBarcodeDLL” (ByVal hBarcode As System.IntPtr , ByVal newValue As Boolean) As Boolean

Visual C++:

```
extern "C" {
BOOL __stdcall mtGetPropertyName (HANDLE hBarcode);
BOOL __stdcall mtSetPropertyName (HANDLE hBarcode , BOOL bNewValue);
}
```

If the property is of type SHORT:

VB.Net:

Private Declare Function mtGet*PropertyName* Lib “SoftekBarcodeDLL” (ByVal hBarcode As System.IntPtr) As Short
Private Declare Function mtSet*PropertyName* Lib “SoftekBarcodeDLL” (ByVal hBarcode As System.IntPtr , ByVal newValue As Short) As Short

Visual C++:

```
extern "C" {
short __stdcall mtGetPropertyName (HANDLE hBarcode);
short __stdcall mtSetPropertyName  (HANDLE hBarcode , short bNewValue);
}
```

If the property is of type STRING:

VB.Net:

Private Declare Function mtGet*PropertyName* Lib “SoftekBarcodeDLL” (ByVal hBarcode As System.IntPtr) As String
Private Declare Function mtSet*PropertyName* Lib “SoftekBarcodeDLL” (ByVal hBarcode As System.IntPtr , ByVal newValue As String) As String

Visual C++:

```
extern "C" {
LPCSTR __stdcall mtGetPropertyName (HANDLE hBarcode);
LPCSTR __stdcall mtSetPropertyName  (HANDLE hBarcode , LPCSTR bNewValue);
}
```

20.2 2DTimeOutPcnt

Overview

2DTimeOutPcnt is the percentage of TimeOut at which the toolkit will stop searching for 2D barcodes. Note that in XML settings this property is known as TwoDTimeOutPcnt.

Type: SHORT
Default Value: 80

See also: TimeOut

20.3 AllowDuplicateValues

Overview

The AllowDuplicateValues can be used to stop the toolkit from reporting duplicate barcodes on the same page in an image. This can be useful for images where the middle section of a barcode is badly damaged or missing. With the property set to TRUE the toolkit may report that there are 2 barcodes of the same type and value. With the property set to FALSE it would assume that the 2 barcodes were part of a single barcode and set the bounding rectangle accordingly.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

MinSeparation

20.4 BarcodesAtTopOfPage

Overview

When true the toolkit will process the scan lines in an image from the top of the page downwards. This is different to ScanDirection, which sets the orientations of barcodes that the toolkit will recognize. It should only be set to True if either MultipleRead is False or MaxBarcodesPerPage is non zero. For other cases it is recommended to leave BarcodesAtTopOfPage at the default value of False.

Type: BOOL
Default Value: FALSE

See also: MultipleRead

MaxBarcodesPerPage

20.5 BitmapResolution

Overview

BitmapResolution is the resolution of the bitmap to be scanned in ScanBarCodeFromBitmap, in dots per inch. This value only effects the expected size of the quiet area around a barcode and for most images can be left to the default value.

Type: SHORT
Default value: 200

See also: Setting and Getting Property Values

20.6 CodabarMaxVariance

Overview

CodabarMaxVariance is the maximum percentage variance that a character in a codabar barcode can have from the average for that barcode.

Type: SHORT
Default value: 20

Note: This is an **Advanced property** and can only be set using LoadXMLSettings

See also: ReadCodabar

20.7 Code128DebugMode

Overview

Code128DebugMode may be used to either ignore the value of a checksum character in a Code 128 barcode or display the values of the Code 128 characters and the calculated checksum value. Note that this is intended for debug analysis of bar codes only.

Possible values:

0 The checksum character must match the value calculated from the rest of the data.
1 The value of the checksum character is ignored.
2 The values of each character are printed in the format shown below.

When set to 2 the following format is used:

Start,Char1,…..,CharN,Checksum(Calculated_Checksum),Stop

e.g 105,30,1,63,35,101,20,98(61),106

Type: SHORT
Default value: 0

See also: Setting and Getting Property Values

20.8 Code128Lenient

Overview

When Code128Lenient is set to True (or 1) the SDK will relax some of the rules for Code 128 bar codes such as the ratio between the pitch of the characters and the spaces between characters.

Type: BOOL
Default value: False or 0

See also: Setting and Getting Property Values

20.9 Code128SearchLevel

Overview

Code128SearchLevel controls the amount of processor time spent on trying to detect Code 128 barcodes. There are 3 levels with 1 being the fastest and 3 the slowest.

Type: SHORT
Default value: 1

See also: Setting and Getting Property Values

20.10 Code25Checksum

Overview

When True the toolkit will only report Code 25 barcodes where the last character is a valid checksum for the rest of the barcode. The toolkit expects a Code 25 checksum to be calculated using the following method:

Sum all of the even positioned characters (the right hand message character is always even), and multiply by 3.

Sum all the odd positioned characters.

Sum the totals from steps 1 and 2.

The checksum is the smallest number that when added to this sum results in a multiple of 10.

If the resulting number of characters is odd and you are using Interleaved Code 2 of 5 then add a leading 0 to the message data.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

ReadCode25

20.11 Code25PitchVariation

Overview

The maximum percentage variation allowed in the width of Code 25 characters.

Type: SHORT
Default value: 20

Note: This is an **Advanced property** and can only be set using LoadXMLSettings

See also: Setting and Getting Property Values

ReadCode25

20.12 Code39Checksum

Overview

When True the toolkit will only report Code 39 barcodes where the last character is a valid checksum for the rest of the barcode. The toolkit expects a Code 39 checksum to be calculated using modulus-43.

The following table shows the character and value used for the calculation...

Char	Value	Char	Value	Char	Value	Char	Value
0	0	B	11	M	22	X	33
1	1	C	12	N	23	Y	34
2	2	D	13	O	24	Z	35
3	3	E	14	P	25	-	36
4	4	F	15	Q	26	.	37
5	5	G	16	R	27	space	38
6	6	H	17	S	28	$	39
7	7	I	18	T	29	/	40
8	8	J	19	U	30	+	41
9	9	K	20	V	31	%	42
A	10	L	21	W	32		

e.g

Data = 12345ABCDE+

Sum of values: 1 + 2 + 3 + 4 + 5 + 10 + 11 + 12 + 13 + 14 + 41 = 116
116 / 43 = 2 rem 30, so U is the check digit.
Data and check digit = 12345ABCDE+U

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

ReadCode39

Code39NeedStartStop

ExtendedCode39

20.13 Code39MaxRatioPcnt

Overview

Code39MaxRatioPcnt is the maximum permitted ratio between the narrow and wide bars or spaces in a Code 39 bar code. For example, a value of 70 means that all the narrow bars and spaces in single character must be no larger than 70% of the size of any wide bar or space. Note that it is not unusual for the scanning process enlarge either the bars or the spaces, which could mean that a valid bar code has narrow space almost as large as a wide bar or vice versa. For this reason this test is only carried out if the property is non-zero.

Type: SHORT
Default value: 0

See also: Setting and Getting Property Values

20.14 Code39NeedStartStop

Overview

When set to TRUE the toolkit will only report Code 39 barcodes that start and end with a * character.

Setting this property to FALSE is not recommended for the following reasons:

It is not a valid Code 39 barcode without the start and stop * character.

Without a start/stop * character, a Code 39 barcode reads with 2 different values, left to right, and right to left. The toolkit will report it as 2 different barcodes unless the scan direction is restricted to one direction only.

The probablility of a false positive reading is increased significantly by setting this property to FALSE.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

20.15 ColorChunks

Overview

ColorChunks specifies how many sections a scan line of an image should be broken into when calculating threshold levels for black and white pixels.

Type: SHORT
Default value: 1

Note: This is an **Advanced property** and can only be set using LoadXMLSettings

See also: ColorProcessingLevel

20.16 ColorProcessingLevel

Overview

The ColorProcessingLevel property controls the amount of processing time spent reading barcode values from color images. Values range from 0 to 5, with a default of 2. A low value will process color images faster but accuracy and read-rate levels will be lower than if a high value is used.

Please note that setting the ColorThreshold property to a non-zero value effectively sets ColorProcessingLevel to 0.

Type: SHORT
Default value: 2

See also: Setting and Getting Property Values

ColorChunks

ColorThreshold

20.17 ColorThreshold

Overview

ColorThreshold is the color value used by the control to decide whether a pixel should be considered to be black or white. The value should be in the range 0 to 255.

Please note that if this property is set to a non-zero value than ColorProcessingLevel is effectively set to a value of 0. It is recommended to set this property to 0 and control the accuracy of reading from color images through the ColorProcessingLevel property.

Type: SHORT
Default value: 0

See also: Setting and Getting Property Values

ColorProcessingLevel

20.18 ConvertUPCEToEAN13

Overview

A UPC-E barcode is actually an EAN-13/UPC-A barcode that has had certain digits removed to create an 8 digit number. Only certain EAN-13/UPC-A barcodes can go through this process. For example, the UPC-A barcode "023456000073 " can be suppressed to the UPC-E value "02345673" and restored to it's original value by the barcode reader. The Softek barcode Reader SDK can interpret a UPC-E barcode in either format via the ConvertUPCEToEAN13 property.

When set to TRUE the toolkit will convert type UPC-E barcodes into EAN-13 format.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

ReadUPCE

ReadEAN13

ReadUPCA

20.19 DatabarOptions

Overview

The DatabarOptions property can be used to set various options for GS1 Databar recognition. All options are turned on by default, but some applications may find it useful to disable certain features for performance reasons.

The property works as a mask and can be constructed from the following values:

1 Read the supplementary 2-D portion if indicated by the linkage flag.

2 Read RSS-14 barcodes

4 Read RSS-14 Stacked barcodes

8 Read RSS-Limited barcodes

16 Read RSS-Expanded barcodes

32 Read RSS-Expanded Stacked barcodes

256 Return the following values for the type of barcode instead of just ”DATABAR”:

DATABAR-RSS-14
DATABAR-RSS-LIMITED
DATABAR-RSS14-STACKED
DATABAR-RSS-EXPANDED
DATABAR-RSS-EXPANDED-STACKED

Type: SHORT
Default value: 255

See also: Setting and Getting Property Values

ReadDatabar

20.20 DataMatrixFinderGapTolerance

Overview

DataMatrixFinderGapTolerance controls the maxiumum size of a gap (measured in pixels) in the L shaped finder pattern of a data matrix bar code. Note that the gap is measured either horizontally or vertically in the image and so the tolerance may be lower in a rotated bar code.

Type: SHORT
Default value: 5

See also: Setting and Getting Property Values

20.21 DataMatrixAutoUTF8

Overview

When DataMatrixAutoUTF8 is TRUE the SDK will not encode data from a DataMatrix that is already encoded in UTF8 format. Note that is it possible for binary data to appear to be valid UTF8 data in which case this property should be set to False.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

20.22 DataMatrixParams

Overview

The DataMatrixParams property controls 3 parameters that affect the recognition of data matrix timer patterns. The property is a string in the following format:

“P1,P2,P3”

where…

P1 = factor of variance for black and white widths in the timer pattern.
P2 = minimum size in pixels of the above variance.
P3 = factor that determines the minimum size of any white and black width in a timer pattern.

Note that this property should be left at default for normal operation and is not exported in XML settings.

Type: STRING
Default value: “4,5,5”

Note: This is an **Advanced property** and can only be set using LoadXMLSettings

20.23 DataMatrixRectangleSupport

Overview

DataMatrixRectangleSupport controls what formats of rectangular data matrix barcode the toolkit will support.

0 = Only read square format data matrix bar codes. This mode improves the speed of data matrix recognition.

1 = Read square and rectangular data matrix bar codes as specified in the ISO/IEC 16022 specification.

2 = Also read extended rectangular data matrix bar codes in the following sizes (DIN16587):

64 X 8, 64 X 12, 64 X 16, 48 X 24, 64 X 24, 40 X 26, 48 X 26, 64 X 26

3 = Also read extended rectangular data matrix bar codes in the following sizes (DIN16587 & ISO/IEC21471):

64 X 8, 64 X 12, 64 X 16, 48 X 24, 64 X 24, 40 X 26, 48 X 26, 64 X 26, 80 X 8, 96 X 8, 120 X 8, 144 X 8, 88 X 12, 36 X 20, 44 X 20, 64 X 20, 48 X 22

Type: SHORT
Default value: 1

20.24 DataMatrixSearchLevel

Overview

DataMatrixSearchLevel can be used to balance speed of recognition against the read rate for data matrix bar codes. Lower values can lead to faster processing where as higher values can result in a better read rate.

Range of values: 1 to 5

Type: SHORT

Default value: 3

20.25 DeskewAfterNormalScan

Overview

Use DeskewAfterNormalScan to force the toolkit to perform a normal scan (one that assumes the bar codes are upright or rotated by 90 degrees) before scanning at other angles.

Type: BOOL

Default value: FALSE

20.26 DeskewMode

Overview

DeskewMode controls how the toolkit handles documents that are skewed or rotated.

Mode	Can be skewed?	Must be aligned?	Max angles
1	No	Yes	1
2	Yes	Yes	1
3	Yes	Yes	2
4	Yes	No	3
5	Yes	No	4
6	Yes	No	5+

Can be skewed? = Barcodes might not be horizontal or vertical.

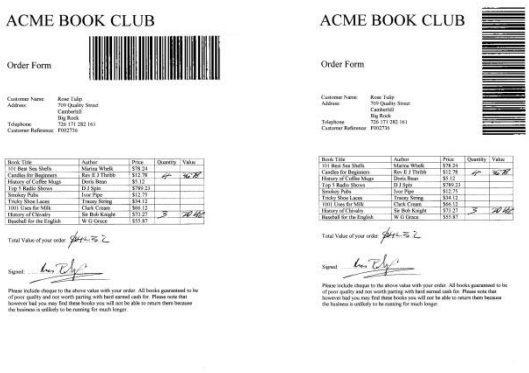
Must be aligned? = Barcodes are aligned if they are parallel or perpendicular (at right angles) to each other.

Max angles = Maximum number of deskew angles. Some complex images may contain multiple possible deskew angles or the barcodes may not be at the same angle as the rest of an image.

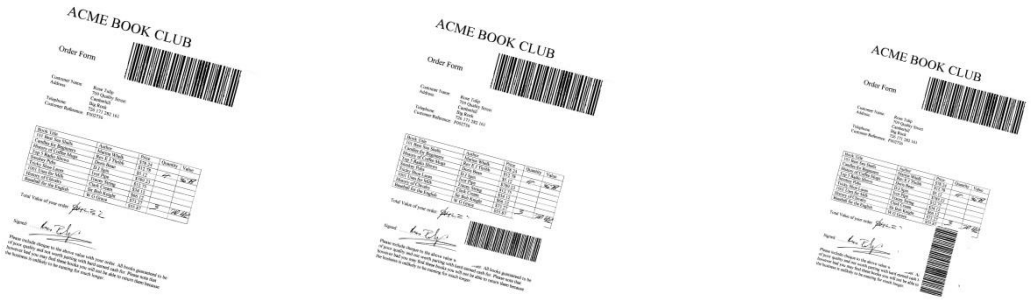
Mode 0 is compatibility mode. In this mode the toolkit behaves as in previous versions with respect to skewed images.

In Modes 1+ the property SkewTolerance has no effect.

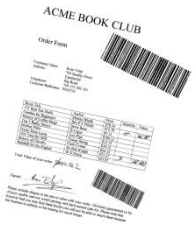
Mode 1 examples (nothing skewed)…



Mode 2 examples (images can be skewed but barcodes always line up with the rest of the image)…



Mode 3 examples (barcodes might not line up with the rest of the image but line up with each other)…



Mode 4+ example (barcodes at increasing numbers of different angles to each other and the base image)…

Type: SHORT
Default value: 2

See also: [Setting and Getting Property Values]

[NoiseReduction]

20.27 Despeckle

Overview

If the **Despeckle** property is set to TRUE and the NoiseReduction property is none zero, then the toolkit removes white speckles inside the bars of a barcode before removing black marks from the spaces between bars.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

NoiseReduction

20.28 DotDataMatrixSupport

Overview

(New in version 8.3.3.5)

When DotDataMatrixSupport is set to 1 the SDK make a specific check for dotted pattern Data Matrix barcodes when the ReadDataMatrix property is also enabled.

Type: SHORT
Default value: 0

Note: This is an **Advanced property** and can only be set using LoadXMLSettings

20.29 EdgeThreshold

Overview

(New in version 8.3.3.5)

EdgeThreshold is the minimum difference in pixel value (when converted to gray scale 0 to 255) for the toolkit to consider the transition as a possible edge in an image. This property was introduced to improve performance when processing large color images, especially when processing DataMatrix barcodes.

A value of 0 gives behavior as in versions prior to 8.3.3.5.

Type: SHORT
Default value: 20

Note: This is an **Advanced property** and can only be set using LoadXMLSettings

20.30 Encoding

Overview

The Encoding property controls the format in which the toolkit returns strings for barcode types that use full symbol sets such as PDF-417.

The property can take any of the following values:

0 Raw with null characters suppressed.

1 Quoted printable

2 Unicode

3 UTF-8

Type: SHORT
Default Value: 3

See also: Setting and Getting Property Values

20.31 ErrorCorrection

Overview

Some barcodes cannot be read because the process of scanning or faxing has split or merged bars together. When ErrorCorrection is set to True to toolkit will, where possible, make a best guess at such barcodes.

Note that this property currently only applies to Code 39 and Code 39 Extended barcodes.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

ReadCode39

20.32 ExtendedCode39

Overview

A Code 39 barcode can be used to represent the entire ASCII-128 symbol set by using 2 normal Code 39 characters to represent one character in the ASCII-128 symbol set. A barcode reader cannot distinguish between normal and extended Code 39 barcodes and so the ExtendedCode39 property must be set to TRUE when reading barcodes encoded using the extended symbol set. Note that the ReadCode39 property must also be set to TRUE.

If the toolkit is unable to decode the string in the extended symbol set then it is left as a normal Code 39 barcode.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

ReadCode39

20.33 FastScanLineJump

Overview

FastScanLineJump sets the frequency at which scan lines are sampled in an image for a fast scan. Whilst the value for LineJump defaults to 1 to ensure that all possible barcodes are detected, FastScanLineJump defaults to 25 in order to quickly capture any easy-to-read barcodes. Using smaller values may result in a decrease in overall speed.

Type: SHORT
Default Value: 25

See also: UseFastScan

LineJump

20.34 FilePathEncoding

Overview

FilePathEncoding specifies how file paths are encoded when passed to DLL or shared object library functions. A value of 0 indicates ASCII encoding and 1 indicates UTF-8 encoding. Wrapper interfaces such as Java and .NET set this property to 1 by default where as the default otherwise is 0.

Type: SHORT
Default value: (see above)

Note: This is an **Advanced property** and can only be set using LoadXMLSettings

20.35 GammaCorrection

Overview

If GammaCorrection is set to a value other than 100 then the toolkit will apply gamma correction to a color image. The amount of gamma correction is equal to GammaCorrection / 100. For example, to achieve a gamma correction of 0.5 the property should be set to a value of 50.

Type: SHORT
Default value: 100

See also: Setting and Getting Property Values

20.36 LicenseKey

Overview

Use the LicenseKey property to set your license key prior to calling the ScanBarCode, ScanBarCodeFromBitmap or ScanBarCodeFromDIB functions. With no license key the .net interface will return all barcode values as “Please contact sales@bardecode.com for a trial license string" and other interfaces will display a pop up box that the user will need to click on to continue.

Type: STRING
Default value: “”

See also: Setting and Getting Property Values

20.37 LineJump

Overview

The LineJump property controls the frequency with which the toolkit samples scan lines as it moves through an image. Increasing the value of the LineJump property will increase the speed at which an image is processed but may decrease the read rate. The SkewLineJump property is used in a similar way when searching for skewed barcodes.

Type: SHORT
Default value: 1

See also: Setting and Getting Property Values

SkewLineJump

20.38 MaxBarcodesPerPage

Overview

MaxBarcodesPerPage sets the maximum number of barcodes that will be read on a single page. Once the toolkit has found this number of barcodes on a page then it stops all further processing of the page and moves on to process the next page. If the maximum number of barcodes expected on a page is known then it is highly recommended that MaxBarcodesPerPage is set since it can give significant improvements in performance. The default value of 0 means no maximum limit to the number of barcodes on a page.

IMPORTANT: MultipleRead must be set to TRUE for MaxBarcodesPerPage to have any effect.

Type: SHORT
Default Value: 0

See also: MultipleRead

20.39 MaxLength

Overview

MaxLength defines the largest length for a barcode string, including checksum characters.

Type: SHORT
Default value: 999

See also: Setting and Getting Property Values

MinLength

20.40 MaxThreads

Overview

MaxThreads defines the maximum number of threads to be used by an instance of the toolkit, excluding the background thread created when ScanBarCodeInBackground is called. When MaxThreads is set to 0 the toolkit will determine the number of threads to use from the number of cores in the processor. The toolkit will not use more threads than the number of cores available in the system.

Type: SHORT
Default value: 4

See also: Setting and Getting Property Values

20.41 MedianFilter

Overview

When TRUE the toolkit will apply a median filter to the image before checking for barcodes. This is a useful option for high resolution images that contain speckles of black and white. It is not recommended for images where the black bars or white spaces are less than 2 pixels wide.

Type: BOOL
Default Value: FALSE

See also: Setting and Getting Property Values

20.42 MedianFilterBias

Overview

Used when applying a median filter to a black and white (2 color) image to control bias given to either black or white. The values can range from 1 to 9 with lower values making an image darker.

Type: SHORT
Default Value: 5

See also: Setting and Getting Property Values

20.43 Min2DLength

Overview

MinLength defines the smallest length for a 2-D barcode.

NOTE: This is currently only changed using LoadXMLSettings.

Type: SHORT
Default value: 1

See also: Setting and Getting Property Values

MaxLength

20.44 MinLength

Overview

MinLength defines the smallest length for a 1-D barcode string, including checksum characters.

Type: SHORT
Default value: 4

See also: Setting and Getting Property Values

MaxLength

Min2DLength

20.45 MinOccurrence

Overview

Please refer to PrefOccurrence for more information.

Type: SHORT
Default value: 2

See also: Setting and Getting Property Values

20.46 MinSeparation

Overview

MinSeparation defines the minimum distance between barcodes of identical value and vertical alignment in 1/300th of an inch. If the distance between two barcodes of same value and on the same alignment is less than MinSeparation then the toolkit assumes that it is a single barcode that has been split into 2 parts by a problem in the scanning process.

Type: SHORT
Default value: 180

See also: Setting and Getting Property Values

20.47 MinSpaceBarWidth

Overview

MinSpaceBarWidth is the minimum acceptable size for a space between the bars in a barcode. When set to a value of 0 the toolkit will automatically select the best value. Spaces that are smaller than the value used are ignored.

Type: SHORT
Default value: 0

See also: Setting and Getting Property Values

20.48 MultipleRead

Overview

Normally the toolkit stops at the first positive match for a barcode. When **MultipleRead** is TRUE the toolkit will check the entire image for barcode strings and record each positive match. When FALSE the toolkit will stop searching as soon as a barcode is found in an image.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

20.49 NoiseReduction

Overview

If the **NoiseReduction** property is none zero then the toolkit will run an image through a noise reduction filter before scanning for barcodes. The filter removes marks from an image that are unlikely to be part of a barcode. A larger value for NoiseReduction will remove larger marks from the image, but may also destroy vital barcode information. A typical value for **NoiseReduction** is 10.

Type: SHORT
Default value: 0

See also: Setting and Getting Property Values

Despeckle

20.50 PageNo

Overview

PageNo is a 1 based index that specifies the page to be scanned in an image. A value of zero indicates that every page will be scanned

Type: SHORT
Default value: 0

See also: Setting and Getting Property Values

20.51 PatchCodeMinOccurrence

Overview

Please refer to PrefOccurrence for more information.

Type: SHORT
Default value: 30

See also: Setting and Getting Property Values

20.52 Pattern

Overview

The Pattern property is a regular expression that each barcode found in an image is compared against. The toolkit will only return barcodes that match the pattern.

The toolkit use POSIX extended regular expression syntax.

Examples:

"ABCDEF" will match all barcodes containing "ABCDEF" (e.g "XYZABCDEFXYZ") .

"ABC[0-9]+" will match all barcodes containing "ABC" followed by one or more digits (e.g XYZABC71827XYZ").

"^ABC[0-9]+$" will match barcodes that only consist of "ABC" followed by one or more digits (e.g "ABC12345").

Note that if a Code 39 barcode uses a checksum character and the Pattern property is used to specify the entire string (ie. the last character of the pattern is $) then the Code39Checksum property must also be set to True.

Type: STRING
Default value: NULL

See also: Setting and Getting Property Values

ReadNumeric

20.53 Pdf417AutoUTF8

Overview

When Pdf417AutoUTF8 is TRUE the SDK will not encode data from a Pdf417 bar code that is already encoded in UTF8 format. Note that is it possible for binary data to appear to be valid UTF8 data in which case this property should be set to False.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

20.54 Pdf417Debug

Overview

Output information about the structure and cluster values of the barcode. The debug information is returned in place of the normal barcode value (see GetBarString).

The fields of the string are as follows:

Error correction status – true or false
Number of data columns
Number of rows
Error correction level
Number of unknown cluster values
And for each cluster: cluster value (score)

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.55 Pdf417ChannelMode

Overview

Pdf417ChannelMode controls whether the toolkit is running is basic or extended channel mode when decoding PDF-417 barcodes. Note that this currently only applies to macro information and does not apply to ECI data.

The symbol identifier is a sequence at the start of the returned data in the format]L*n* where *n* is a digit in the range 0 to 3. E.g]L1

Macro information can be encoded in a PDF-417 barcode and is output following the symbol identifier and before tha data. The sequence \MI is followed by the segment index; \MF is followed by the file id; \MO*n* is an optional field followed by data from the table below;\MZ indicates the end of the segment sequence;\MY indicates the end of the macro information.

Optional fields:

0	File name
1	Segment count
2	Time stamp
3	Sender
4	Addressee
5	File size
6	Checksum

Macro example:

]L1\MI00000\MF123123123\MO100004\MO0somefilename\MO20000001326\MO3John\MO4Simon\MO50000000424\MO725361\MY

Pdf417 Channel Modes:

	Symbol Identifier	\ chars doubled in data	Macro information output	ECI Support
0	Not output	No	No	No
1	]L1	Depends on value of Pdf417MacroEscapeBackSlash	Yes	Not at present
2	]L2	No	No	No
3	]L0	No	No	No

Type: SHORT
Default value: 0

See also: Setting and Getting Property Values

20.56 Pdf417MacroEscapeBackSlash

Overview

When Pdf417ChannelMode is set to 1 the Pdf417MacroEscapeBackslash property controls whether back-slash characters doubled in the data portion of the output.

Type: BOOL
Default: TRUE

See also: Setting and Getting Property Values

20.57 PdfBpp

Overview

PdfBpp is the number of bits-per-pixel to be used when converting a pdf file into image format, before scanning for a barcode. The value should be 1, 8 or 24.

Type: SHORT
Default value: 8

See also: Setting and Getting Property Values

PdfDpi

PdfImageOnly

20.58 PdfDpi

Overview

PdfDpi is the number of dots-per-inch to be used when converting a pdf file into image format, before scanning for a barcode.

Type: SHORT
Default value: 300

See also: Setting and Getting Property Values

PdfBpp

PdfImageOnly

20.59 PdfImageExtractOptions

Overview

This property is no longer used.

Type: SHORT
Default value: 0

See also: PdfImageOnly

20.60 PdfiumPath

Overview

Use PdfiumPath to specify the location of the Pdfium dll or shared object file that the toolkit should use when handling PDF documents. The toolkit will also search for the Pdfium library in the following sequence:

On Windows: Current folder, a folder called DLL within the current folder and all folders on PATH

On Linux and OSX: Current folder, a folder called lib with the current folder, a folder called lib within the parent folder and all folders on LD_LIBARRY_PATH

Note that the toolkit download includes the Pdfium standard library for Windows, Linux and OSX.

Type: SHORT
Default value: 0

20.62 PdfImageOnly

Overview

PdfImageOnly indicates that the PDF documents to be processed by the toolkit are images. Image only PDF documents can be processed faster than other types (containing text, vector graphics etc). If the number of images extracted from a PDF document does not match the number of pages in the document then the document is re-processed as if PdfImageOnly was set to false.

Type: BOOL
Default value: True

Note: In some interfaces this property must be set as an **Advanced Property** using LoadXMLSettings

See also: PdfImageExtractOptions

Setting and Getting Property Values

20.63 PdfImageRasterOptions

Overview

This property is no longer used.

Type: SHORT
Default value: 0

See also: PdfImageOnly

20.64 PdfLocking

Overview

This property is no longer used.

Type: SHORT
Default value: 2

See also: Setting and Getting Property Values

20.65 PdfPassword

Overview

The password for opening PDF documents.

Type: STRING
Default value: “”

20.66 Photometric

Overview

The Photometric property determines how the toolkit interprets a pixel value in a bi-tonal bitmap passed to the ScanBarCodeFromBitmap method.

	Pixel Value = 0	Pixel Value = 1
Photometric = 0 or false	White	Black
Photometric = 1 or true	Black	White

This property is not used with the ScanBarCode or ScanBarCodeFromDIB methods.

Type: SHORT or BOOL depending on interface
Default value: 0 or False in all interfaces except for .Net where the default is True.

See also: Setting and Getting Property Values

ScanBarCodeFromBitmap

20.67 PrefOccurrence

Overview

As the SDK scans an image it assigns a score to each barcode candidate. At the end of a scan, any candidates with a score >= PrefOccurrence are reported by the SDK. If no candidate meets this criteria then the SDK selects the candidate with the highest score and reports this barcode if it has a score >= MinOccurrence. Note that Patch Codes are only ever reported if the score is >= PatchCodeMinoccurrence.

Type: SHORT
Default value: 5

See also: Setting and Getting Property Values

MinOccurrence

PatchCodeMinOccurrence

20.68 QRCodeAutoMedianFilter

Overview

When QRCodeBWAutoMedianFilter is TRUE the SDK will automatically apply a median filter to a black and white image if it is unable to locate a QR-Code without using a median filter. It will additionally set MedianFilterBias to a value of 1 when applying the median filter to darken the image.

Type: BOOL
Default value: TRUE

20.69 QRCodeAutoUTF8

Overview

When QRCodeAutoUTF8 is TRUE the SDK will not encode data from a QrCode that is already encoded in UTF8 format. Note that is it possible for binary data to appear to be valid UTF8 data in which case this property should be set to False.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

20.70 QRCodeByteMode

Overview

QRCodeByteMode controls the way in which byte compaction sequences are handled in byte segments within QR Code data.

Byte segments are a sequence of values in the range 0 to 255 and can be used to store any kind of data. However, the ISO specification states that certain pairs of bytes (compaction sequences) should be interpreted as Kanji characters even though such sequences can randomly occur in other types of data. The QRCodeByteMode property allows applications to control how this ambiguity should be handled and also expands the supported set of compaction sequences to include Big5 and GBK.

In practice the data within a byte segment tends to contain binary data, ASCII text, UTF-8 or a mixture of Chinese characters and ASCII text. The Softek SDK will try to automatically figure out the encoding method used in a byte segment but since there is nothing within a QRCode to say for sure how the data was encoded it may be necessary to set QRCodeByteMode to specific value in order to get the correct results for an application.

There are 3 types of byte compaction:

- Kanji - used for encoding the Chinese characters adopted in the Japanese writing system.
- Big5 – used for encoding traditional Chinese characters in Taiwan, Hong Kong and Macau.
- GBK – used for encoding simplified Chinese characters in the People's Republic of China.

Note:

- Byte segments may also be used to store data in UTF-8 format or in binary format. UTF-8 is not considered to be byte compaction for the purposes of this description.
- Kanji byte compaction is the only encoding scheme that is part of the ISO specification for QR-Codes.
- Byte compaction sequences are not unique to one scheme, so the value for QRCodeByteMode is application dependent.
- Sequences that decode as Big5 data will also decode as GBK data.

The possible values and order of decoding are as follows:

Value	Comment	Order
0	No byte compaction support	UTF-8(*), binary
1	Auto-detect (GBK before Big5)	Kanji, UTF-8(*), GBK, Big5, Binary
2 - default	Kanji byte compaction	Kanji, UTF-8(*), Binary
3	Big5 byte compaction	Big5, UTF-8(*), Binary
4	GBK byte compaction	GBK, UTF-8(*), Binary
5	Auto-detect (Big5 before GBK)	Kanji, UTF-8(*), Big5, GBK, Binary

(*) UTF-8 detection only occurs when QRCodeAutoUTF8 is enabled.

The following values are also available where byte compaction sequences are mixed with other 8-bit values:

Value	Comment	Order
6	Mixed Kanji/Binary	Kanji/Binary
7	Mixed Big5/Binary	Big5/Binary
8	Mixed GBK/Binary	GBK/Binary

Type: SHORT
Default value: 2

Note: This property must be set as an **Advanced Property** using LoadXMLSettings

See also: Setting and Getting Property Values

20.71 QRCodeKanjiModeConvertUTF8

Overview

QRCodeKanjiModeConvertUTF8 controls how the SDK processing Kanji shift jis codes.

A value of 1 means that the SDK converts all shift jis codes in Kanji mode to UTF8. This is the default.

A value of 0 means that the SDK does not convert shift jis codes to UTF8. The codes are added to the output as if they were binary byte data.

Note: This property must be set as an **Advanced Property** using LoadXMLSettings

Type: SHORT
Default value: 1

20.72 QRCodeReadInverted

Overview

When QrCodeReadInverted is TRUE the SDK will automatically scan for QrCodes that are white on a black background (in addition to the normal black on a white background).

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

20.73 QuietZoneSize

Overview

When the toolkit checks for a barcode on a scan line in an image, it ignores those parts of the line that are not preceded by the number of white pixels specified by QuietZoneSize. When the property has a value of 0 then the quiet zone is calculated 1/10th of the value of the image resolution (e.g. 10 pixels in a 100 dpi image).

Type: SHORT
Default value: 0

See also: Setting and Getting Property Values

20.74 QuotedPrintableCharSet

Overview

Defines the character set to be used when using quoted printable encoding. Note that in ASCII mode multi-byte characters will be omitted from barcode values.

0 = ASCII

1 = UTF-8

Type: INTEGER
Default value: 0

See also: Setting and Getting Property Values

20.75 ReadCodabar

Overview

When set to TRUE the toolkit will search for codabar barcodes and the string returned by GetBarStringType will be set to “CODABAR”.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

CodabarMaxVariance

20.76 ReadCode128

Overview

When set to TRUE the toolkit will search for type 128 barcodes and the string returned by GetBarStringType will be set to CODE128.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

ReadShortCode128

UseOldCode128Algorithm

20.77 ReadCode25

Overview

When set to TRUE the toolkit will search for type 2 of 5 interleaved barcodes and the string returned by GetBarStringType will be set to “CODE25”.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

ReadCode25ni

Code25Checksum

20.78 ReadCode25ni

Overview

When set to TRUE the toolkit will search for type 2 of 5 non-interleaved barcodes in the following formats:

- Code 2 of 5 Datalogic
- Code 2 of 5 Iata1
- Code 2 of 5 Iata2
- Code 2 of 5 Industrial
- Code 2 of 5 Interleaved
- Code 2 of 5 Matrix

The string returned by GetBarStringType will be set to “CODE25”.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.79 ReadCode39

Overview

When set to TRUE the toolkit will search for type 39 barcodes and the string returned by GetBarStringType will be set to “CODE39”.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

Code39Checksum

Code39NeedStartStop

ExtendedCode39

20.80 ReadCode93

Overview

When set to TRUE the toolkit will search for type 93 barcodes and the string returned by GetBarStringType will be set to “CODE93”.

Type: BOOL
Default value: FALSE

Note: In some interfaces this property must be set as an **Advanced Property** using LoadXMLSettings

See also: Setting and Getting Property Values

20.81 ReadDatabar

Overview

When set to TRUE the toolkit will search for GS1 Databar barcodes and the string returned by GetBarStringType will be set to “DATABAR”. The following types of GS1 Databar are supported:

RSS-14

RSS-14 Truncated

RSS-14 Stacked

RSS-14 Stacked Omnidirectional

RSS Limited

RSS Expanded

RSS Expanded Stacked

Please note the the bounding rectangle for stacked versions of the barcode currently only includes either the top-most or bottom-most element of the stack.

Reading supplementary data

Some GS1 Databar barcodes encode supplementary data in the form of a micro-PDF-417 barcode above the linear portion of the barcode. To read the supplementary portion set ReadMicroPDF417 to True and ensure that DatabarOptions includes the option to read supplementary barcodes.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

ReadMicroPDF417

DatabarOptions

20.82 ReadDataMatrix

Overview

When set to TRUE the toolkit will search for DataMatrix (ECC 200) barcodes and the string returned by GetBarStringType will be set to “DATAMATRIX”.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.83 ReadEAN13

Overview

When set to TRUE the toolkit will search for EAN-13 type barcodes and the string returned by GetBarStringType will be set to “EAN13”.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

20.84 ReadEAN13Supplemental

Overview

When set to TRUE (and ReadEAN13 is also set to TRUE) the toolkit will search for EAN-13 supplemental information. If a supplemental EAN bar code is found then it is appended to the bar code string after a space.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.85 ReadEAN8

Overview

When set to TRUE the toolkit will search for EAN-8 type barcodes and the string returned by GetBarStringType will be set to “EAN8”.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

20.86 ReadMicroPDF417

Overview

When set to TRUE the toolkit will search for micro-PDF-417 barcodes and the string returned by GetBarStringType will be set to “PDF417”.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.87 ReadNumeric

Overview

When True the toolkit will only report numeric barcodes. Note that this the same as setting the Pattern property to the value "^[0-9]+$".

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.88 ReadPatchCodes

Overview

When set to TRUE the toolkit will search for patch code barcodes and the string returned by GetBarString will be set to PATCH.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.89 ReadPDF417

Overview

When set to TRUE the toolkit will search for PDF-417 barcodes and the string returned by GetBarString will be set to “PDF417”.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.90 ReadQRCode

Overview

When set to TRUE the toolkit will search for QR-Codes (model 2) and the string returned by GetBarString will be set to “QRCODE”.

- All version sizes are supported, however best results will always be obtained when using smaller version sizes with maximum error correction.
- Skewed QR-Codes can be read using the default value for SkewTolerance. Changing the value of SkewTolerance will have no effect on the scanning of QR-Codes.
- GetBarStringDirection will always return a value of 1 for QR-Codes.
- If only QR-Codes need to be recognized then set ScanDirection to a value of 1.
- There is no support for QRCode model 1.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.91 ReadShortCode128

Overview

When set to TRUE the toolkit will search for Code 128 barcodes of symbol set C, without the normal start and stop characters. The barcode type for these barcodes is set to "SHORTCODE128".

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.92 ReadUPCA

Overview

When set to TRUE the toolkit will search for UPC-A type barcodes.

UPC-A barcodes are a subset of EAN-13. For example, the UPC-A barcode "016000336100" is the same as the EAN-13 barcode "0016000336100". In fact, UPC-A barcodes are the sub-set of EAN-13 barcodes that start with a 0. The ReadEAN13 property controls whether any barcodes of type EAN-13 are recognized - and this includes UPC-A, whether or not ReadUPCA is set to true. The effect of the ReadUPCA flag is to control whether an EAN-13 barcode that starts with a 0 is returned as a 12 digit UPC-A or as a 13 digit EAN-13 barcode. The string returned by GetBarString will be set to either “UPCA” or “EAN13”.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.93 ReadUPCE

Overview

When set to TRUE the toolkit will search for UPC-E type barcodes.

A UPC-E barcode is actually an EAN-13/UPC-A barcode that has had certain digits removed to create an 8 digit number. Only certain EAN-13/UPC-A barcodes can go through this process.

For example, the UPC-A barcode "023456000073" can be suppressed to the UPC-E value "02345673" and restored to its original value by the barcode reader. The Softek barcode Reader SDK can interpret a UPC-E barcode in either format via the ConvertUPCEToEAN13 property. The string returned by GetBarStringType will be set to “UPCE”.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.94 ReportUnreadBarcodes

Overview

ReportUnreadBarcodes is a mask that controls the reporting of barcodes that could not otherwise be successfully decoded. Please note the following limitations:

- Maximum of 1 unread barcode reported per page.
- No unread barcodes reported on a page where another barcode was successfully read.
- Linear barcodes must consist of at least 40 black bars.
- 2-D barcodes will only be reported if they reach and fail the error correction phase of decoding.
- Any large series of parallel lines will be reported as an unread linear barcode.
- All unread barcodes are reported as type “UNREAD” with value “UNREAD”.
- The appropriate 2-D type must be enabled to detect unread barcodes of that type.
- If a linear barcode type is disabled then barcodes of that type may be reported as unread.

Mask values:

1 = Linear barcodes
2 = Datamatrix
4 = QR-Code
8 = PDF-417

Type: SHORT
Default Value: 0

20.95 RescaleTiffAllowed

Overview

RescaleTiffAllowed controls whether TIFF documents with differing horizontal and vertical resolutions should be rescaled to match the horizontal resolution. This can effect the recognition of certain 2-D barcodes.

Type: BOOL
Default Value: TRUE

20.96 RotateBy45IfNoBarcode

Overview

When RotateBy45IfNoBarcode is TRUE the barcode reader will rotate an image by 45 degrees and rescan if no bar code was found in normal orientation.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.97 ScanDirection

Overview

ScanDirection is a mask that controls the directions in which the barcode reader will look for barcodes in an image, and is built from the following values:

1 = Left to Right

2 = Top to Bottom

4 = Right to Left

8 = Bottom to Top

For example, a value of 5 (1 + 4) means that the reader will look for barcode from left to right and right to left.

Note: This property replaces the Rotation property used in previous versions.

Type: SHORT
Default value: 15

See also: Setting and Getting Property Values

20.98 ShortCode128MinLength

Overview

ShortCode128MinLength defines the smallest length for a barcode string, including checksum characters.

Type: SHORT
Default value: 2

See also: Setting and Getting Property Values

20.99 Show2DCornersInResults

Overview

Include the corners coordinates of Datamatrix and QRCode barcodes with the barcode string. The coordinates are appended to the string in the format:

@ upper right x, upper right y, lower left x, lower left y, upper left x, upper left y

e.g

@539,1973,388,2116,391,1970

Type: SHORT
Default value: 0

Note that this is an advanced property and can only be set using LoadXMLSettings.

20.100 ShowCodabarStartStop

Overview

Include the start and stop characters when returning the value of a codabar barcode.

Type: SHORT

Default value: 1

Value	Output
0	Do not output start/stop characters
1	Display a, b, c or d at the start and t, n, * or e at the end
2	Display A, B, C or D at the start and T, N, * or E at the end
3	Display A, B, C or D at the start and the end

20.101 ShowCheckDigit

Overview

When set to TRUE the toolkit will include the barcode check digit in the returned string.

Note: This property only applies to barcode types with built in check digits (e.g Code 128).

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.102 SkewedDatamatrix

Overview

If SkewedDatamatrix and ReadDatamatrix are both set to True then the toolkit will search for skewed datamatrix barcodes even when SkewTolerance is at the default value of 0.

Type: BOOL
Default Value: FALSE

See also: SkewedLinear

SkewTolerance

Setting and Getting Property Values

20.103 SkewedLinear

Overview

When True the toolkit will search for skewed linear barcodes even when SkewTolerance is at the default value of 0. Note that currently this property only relates to Codabar, Code 25, Code 39 and Code 128 barcodes.

Type: BOOL
Default Value: TRUE

See also: SkewedDataMatrix

SkewTolerance

Setting and Getting Property Values

20.104 SkewLineJump

Overview

SkewLineJump works in a similar way to the LineJump property, but only effects the phase of the scanning process concerned with searching for skewed barcodes. It can be useful to set the 2 properties to different values for reasons of performance.

Type: SHORT
Default value: 9

See also: Setting and Getting Property Values

20.105 SkewTolerance

Overview

IMPORTANT: From version 8.3.1.1 onwards this setting only has effect is DeskewMode has the value 0.

Many skewed barcodes can (since version 7.6.1) be recognized by the toolkit using default settings, however some examples may still require the use of the SkewTolerance setting. SkewTolerance controls the maximum angle from the horizontal or vertical at which a barcode will be recognised by the toolkit. The table below shows the possible values for this property along with the approximate maximum angles:

0 = up to 5 degrees

1 = 13 degrees

2 = 21 degrees

3 = 29 degrees

4 = 37 degrees

5 = 45 degrees

Type: SHORT
Default value: 0

See also: Setting and Getting Property Values

20.106 TifSplitMode

Overview

TifSplitMode controls the way in which multi-page TIF files are split into sub-files when the TifSplitPath property is set.

- 0: Sub-files will each start on a page with a bar code (or page 1).
- 1: Sub-files will terminate on a page with a barcode (or the last page).
- 2: Sub-files consists of the pages between (but not including) the barcodes.
- 3: Sub-files consists of only the pages containing barcodes.
- 4: Sub-files will each start on a page with a different bar code value (or page 1).

For example, a 6 page TIF file with barcodes on pages 2 and 5 will split as follows:

TifSplitMode = 0: First sub-file contains page 1. Second sub-file contains pages 2, 3 and 4. Third sub-file contains pages 5 and 6.

TifSplitMode = 1: First sub-file contains pages 1 and 2. Second sub-file contains pages 3, 4 and 5. Third sub-file contains page 6.

TifSplitMode = 2: First sub-file contains page 1. Second sub-file contains pages 3 and 4. Third sub-file contains page 6.

Type: SHORT
Default value: 0

See also: Setting and Getting Property Values

20.107 TifSplitPath

Overview

TifSplitPath is the file path template for splitting multi-page TIF files into sub-files in either TIF or PDF format. The template can include the following conversion specifications:

%d	replaced by a sub-file index number (starting at 1)
%s	replaced by the barcode found in the sub-file (modes 0 and 1) or the last barcode found before the sub-file (mode 2). If no relevant barcode is available then an empty string is used.

If the template does not include %d or %s, or %s then a %d is assumed before the file extension. This case also applies if %s is used on it's own and there is no relevant barcode value available.

NOTE: The .Net 1 interface cannot output files in PDF format and the .Net 2/3.5 interface requires the PDF Extension to output files in PDF format.

The mode for splitting the TIF file is controlled by the TifSplitMode property.

e.g c:\tmp\output%d_%s.tif

or c:\tmp\output%d_%s.pdf

Type: STRING
Default value: NULL

See also: Setting and Getting Property Values

20.108 TimeOut

Overview

TimeOut is the maximum time in milliseconds that the toolkit will allow for scanning a page in a document. Note that this does not include the time to load a page into memory and is approximate.

Type: Integer
Default Value: 5000

See Also: 2DTimeOutPcnt

Setting and Getting Property Values

20.109 UseFastScan

Overview

When True the toolkit will perform a fast scan of a page before conducting the normal scan. The fast scan is only used when either MultipleRead is set to false or MaxBarcodesPerPage is set to 1 i.e. it is only used when a single barcode is required on a page.

Note:

- The height of bounding rectangles may differ between barcodes discovered using fast and normal scans.
- On a page with multiple barcodes the single barcode selected through a fast scan may differ to that of a normal scan.
- Fast scans are not carried out when datamatrix barcode types are enabled

Type: BOOL
Default Value: TRUE

See Also: FastScanLineJump

Setting and Getting Property Values

20.110 UseOldCode128Algorithm

Overview

Use the Code 128 detection algorithm as used in earlier versions of the toolkit (pre version 7.3.1).

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.111 UseOverSampling

Overview

When UseOverSampling is TRUE the barcode reader samples 3 lines at a time (skipping 2 lines between each sample) and takes the average pixel value. This is useful for images containing both black and white speckles.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

20.112 UseOld1DAlgorithms

Overview

Use the old algorithms for detecting 1-D barcodes (in version 9.1 this applies to only Code 128 and Code 39 barcodes).

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

Note that this is an advanced property and can only be set using LoadXMLSettings.

20.113 UsePrePageLoadTimer

Overview

When checking for TimeOut set the start time to the point just before the page was decompressed and loaded into memory rather than just before the scan of the resultant bitmap.

Type: BOOL
Default value: FALSE

See also: Setting and Getting Property Values

Note that this is an advanced property and can only be set using LoadXMLSettings.

20.114 UseRunCache

Overview

Use a memory cache for run-length information derived from an image.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

20.115 WeightLongerBarcodes

Overview

When WeightLongerBarcodes is TRUE the barcode reader will weight the counts used with the PrefOccurrence and MinOccurrence properties according to the length and type of the barcode in question. Barcode types using built in checksums are favoured above barcode types with no checksum.

Type: BOOL
Default value: TRUE

See also: Setting and Getting Property Values

21 Appendix C: Installation Files

This section shows the dll files required for the various interfaces to the toolkit. Note that one approach is to leave all the dll files in the folder created by the install set and either modify the system PATH or specify the location of the files in the application. The section on installation on Production Systems contains details of the options.

21.1 Windows x86 Applications

The following table shows the files necessary to support Windows x86 based applications. Note that all of these files can either be left in the original installation folder or put in one of the following target folders:

- Windows system32 folder on x86 based systems
- Windows SysWow64 folder on x64 based systems
- Any folder on the applications path

File	Win32 DLL	Java	ActiveX/ OCX	COM	PDF Extension
SoftekBarcodeDLL.dll	✓	✓	✓	✓	
SoftekBarcodeCOM.dll				✓	
SoftekBarcode.ocx			✓		
Pdfium.dll					✓

Register with regsvr32.

Example: The COM interface requires the files SoftekBarcodeDLL.dll and SoftekBarcodeCOM.dll, and SoftekBarcodeCOM.dll needs to be registered.

21.2 Windows x64 Applications

The following table shows the files that need to be installed to support x64 based applications. One approach is to leave all the dll files in the original installation folder, but if this isn't appropriate then the target folder for the x64 files can be either the windows system32 folder or any folder on the applications path.

File	Win32 DLL	Java	ActiveX/ OCX	COM	PDF Extension
SoftekBarcode64DLL.dll	✓	✓	✓	✓	
SoftekBarcode64COM.dll				✓	
Pdfium64.dll					✓

Register with regsvr32.

21.3 Linux

The following table shows the files that need to be installed to support Linux based applications somewhere on the LD_LIBRARY_PATH

File	Win32 DLL	Java	PDF Extension
libbardecode.so	✓	✓	
libpdfium.so			✓

21.4 OSX

The following table shows the files that need to be installed to support OSX based applications somewhere on the LD_LIBRARY_PATH

File	Win32 DLL	Java	PDF Extension
libbardecode.dylib	✓	✓	
libpdfium.dylib			✓

21.5 .Net

.Net applications should use the Nuget package which installs all of the required files necessary for operation of the toolkit.

22 Appendix D SoftekSDKDemo.exe Command Line Options

SoftekSDKDemo.exe [/X xml_file] [/C “caption”] [image_file [/R]]

/X xml_file	Load settings from the specified xml file
/C caption	Set the caption for the main SoftekSDKDemo window. Use the underscore character (_) to include a space in the caption.
Image_file	Load the specified image on launch
/R	Automatically scan page 1 of image for barcodes

23 Appendix E: Release Notes

23.1 Version 9.3.1.11

A potential crash when creating PDF documents using Pdfium has been fixed.

23.2 Version 9.3.1.10

Support for image extraction from PDF documents added for images using the ICC color space.

Correction of possible data overrun in reed solomon error correction.

The permitted range of ratio between the sides of a QrCode has been increased to a maximum of 1:2

Fixed corruption of split PDF output files.

Fixed memory leak in PDF417 detection.

Fixed memory leak in DataMatrix detection when a scan time out occurs.

The MinLength property now works correctly with Codabar barcodes.

23.3 Version 9.3.1.7

Improvement to Codabar recognition for higher resolution images where the barcode is close to the edge of the image.

Improvement to the detection of inverted color datamatrix barcodes close to the edge of an image.

Downgrade of zlib to version 1.2.13 (Windows only) to avoid compatibility issues with libpng.

23.4 Version 9.3.1.6

Code 25 – decrease in chance of false positive result in barcodes with more than 3 pixels per module.

Datamatrix – better detection of barcodes with varying spacing in timer patterns.

Reduction in possibility of a false positive datamatrix codes in any of the following cases:

- DataMatrixSearchLevel = 5
- DotDataMatrix = 1 or true
- The size of the grid is 10x10

Zlib version upgraded to 1.3

In the Linux version of the SDK the functions mtScanBarCodeFromBitmap and STReadBarCodeFromBitmap can both now accept bitmaps where bmWidthBytes is not a multiple of 4.

23.5 Version 9.3.1.1

Switch to using Pdfium for reading barcodes from PDF documents.

Improved accuracy for the values returned by GetBarStringPos.

If a PDF417 barcode contains a symbol length descriptor that exceeds the data byte count then it is automatically reduced to this limit.

Reduction in the possibility of duplicate barcodes when DeskewMode >= 3

Improvement to the detection of rotated higher resolution datamatrix barcodes.

Fix for potential data over-run when detecting split or merged bars in a 1D barcode.

BMP files are no longer held open in the Windows DLL.

Support for QRCode Model 1

Support for some QR Codes with one missing finder pattern

Change in the shift jis conversion table to cp932

Addition of the QRCodeKanjiModeConvertUTF8 property

Fix for GetBarStringDirection for barcodes that require rotation

Fixed potential jvm crash creating the Barcode class

Code 39 – reduction in possibility of false positives for color images.

23.6 Version 9.2.3.5

Fixed memory leak in the Code 25 algorithm present since version 9.1.5

23.7 Version 9.2.3.4

Improvement to the reading of dot datamatrix barcodes.

Whenreading QrCodes; Higher levels of ColorProcessingLevel should now produce consistently better results.

Improvement in reading images containing multiple QrCodes.

Improvement in reading a URL encoded in a QrCode.

MaxBarCodesPerPage is now handled correctly when reading DataMatrix barcodes. For example, if an image contained 20 datamatrix barcodes and the software returned 20 barcodes with MaxBarCodesPerPage = 0 then it may have returned a value <20 when MaxBarCodesPerPage = 20.

Correction to photometric interpretation in 1bpp BMP files.

Addition of advanced property UsePrePageLoadTimer

DataMatrix barcodes are now only compared against Min2DLength rather than both Minlength and Min2DLength.

23.8 Version 9.2.2.1

Timeout checks added during image rotation.

Rectangles for poor quality barcodes in rotated images correctly mapped onto original image.

Potential malloc error fixed when reading QrCodes.

23.9 Version 9.1.5.8

The main SDK has been rebuilt using VS2022 optimization and in tests runs about 5% faster than previous versions.

SoftekSDKDemo now handles the ReadQRCode property correctly when editing an XML file.

Improvement to the recognition of QrCodes that have part of an edge missing.

Quiet zones for QrCodes are now enforced correctly for color images larger than 1MB in size.

Fix for very slow QrCode recognition on 64-bit systems with certain images.

Fix for potential memory over-run when reading certain PDF-417 barcodes.

The types for non-interleaved Code 25 barcodes are now returned as follows:

Industrial = "CODE25-IND"

Iata = "CODE25-IATA"

Matrix = "CODE25-MATRIX"

Datalog = "CODE25-DLOG"

23.10 Version 9.1.5.7

Fix for hang when processing QR Codes on 64-bit systems

Improvement to DataMatrix recognition and Dot DataMatrix recognition.

Images whose median gray scale color value is within EdgeThreshold of either 0 or 255 are now considered to be blank.

23.11 Update to documentation

Documentation updated to include 2 additional TifSplitModes:

3: Sub-files consists of only the pages containing barcodes.

4: Sub-files will each start on a page with a different bar code value (or page 1)

23.12 Version 9.1.5.3

Zlib updated to version 1.2.12

23.13 Version 9.1.5.2

Improvement to Code 25 recognition algorithm to reduce the possibility of a false positive result when a barcode has a split bar or a vertical line bisecting the barcode.

The SDK can now handle square datamatrix barcodes with an aspect ratio of up to 1:1.5

Improvement to the datamatrix algorithm to better cope with lines or boxes that are very close to the barcode.

Improvement to the QRCode algorithm in both speed and read rate.

Fix for potential infinite loop when detecting Qr Codes.

23.14 Version 9.1.4.1

The requirement for a minimum distance between 2 identical 2-D barcodes has been removed.

A new property called Min2DLength (minimum length of a 2-D barcodes) has been added with a default of 1.

The default value for MaxLength (maximum barcodes length) has been increased from 999 to 9999.

Code 128 recognition has been improved where there are unusual relative widths between bars and spaces.

QR-Code recognition improved where the finder patterns are damaged.

The value returned by GetBarStringDirection for DataMatrix barcodes has been fixed so it represents the actual orientation of the barcode.

A buffer over-run for huge black and white images containing complex patterns has been fixed.

The speed of recognition of structured QR Codes has been improved.

Code 39 barcodes without start/stop characters are now read as in version 8.

23.15 Version 9.1.3.1

The way the pitch has been measured in barcodes with inter-character separators has been changed so that the pitch is now measured from the left of the first bar to the right of the separator space.

An unintended effect of noise reduction for color images has been fixed.

Fixed problem in Java interface where by character values > 127 were not encoded when using quoted printable encoding (Encoding = 1).

Added UTF-8 validation in the Java interface GetBarString method.

Fixed problem in the QR Code decoder where an invalid sequence could be decoded as garbage data rather than being rejected.

A build option error in the 64-bit version that broke pattern matching has been fixed.

23.16 Version 9.1.2.1

Added ability to read DataMatrix rectangular barcodes as per ISO/IEC21471 by setting DataMatrixRectangleSupport to a value of 3.

Fixed reading of barcodes from GIF files in the x64 version of the DLL.

23.17 Version 9.1.1.10

Improved datamatrix recognition to handle images where the image is cropped to the exact rectangle of the barcode.

23.18 Version 9.1.1.9

Fixed crash when using multiple property sets from an XML file.

23.19 Version 9.1.1.8

Fix for problem of getting licensing error when processing PDF files in concurrent multiple threads.

23.20 Version 9.1.1.7

Fix for calculating the bounding recatangle around a barcode
Fix for correctly reading the color map in a 1 bpp bmp file
Fix for crash on 64-bit systems when QRCode reading and median filter are enabled at the same time.

23.21 Version 9.1.1.6

Fixed potential crash when median filter is used.

23.22 Version 9.1.1.5

Improvement to the recognition of Code 128 barcodes with a large module size but unusually small single module gaps between bars.

Improvement to the detection of marginal skewed parallel barcodes.

Split PDF files with paths that evaluate to greater than 260 characters are now created correctly.

This version is an update to the SoftekBarcodeDLL.dll and SoftekBarcode64DLL.dll files and includes the following 2 changes:

A potential very large memory leak when handling PDF files has been fixed. The potential for the leak has existed in all version of the SDK that use the Debenu DLL files and was most likely to occur with the 64-bit versions of the SDK.

A small change to Code 25 recognition that improves the read rate for codes where the narrow:wide ratio can vary from 1:4 to 2:3.

23.23 Version 9.1.1.1

Support added for reading from TIF/PDF/JPG/GIF/PNG images held in strings or byte arrays. In the .Net interface a new function called ScanBarCodeFromByteArray has been added and in other interfaces the new function is called ScanBarCodeFromString. This means that images held in memory no longer need to be in uncompressed bitmap format and conversely PDF, TIF etc no longer

need to be held as files on disc in order to be used with the SDK. Background versions of the same functions are also available in the form of ScanBarCodeFromStringInBackground and ScanBarCodeFromByteArrayInBackground.

The Debenu PDF DLL files have also been updated to version 1811 and a slightly faster method is now used to extract images or render pages. This change also resolves an issue encountered in Visual Studio 2019 using .Net 5 where by the render process for certain files appeared to cause massive memory usage.

New algorithms have been developed for Code 39 and Code 128 recognition. The old algorithms can still be used if needs be by setting the UseOld1DAlgorithms property.

TIFF documents with differening horizontal and vertical resolutions are now re-scaled to match the horizontal resolution. This can be prevented by setting the property RescaleTiffAllowed to false or 0.

23.24 Version 8.4.1.2

Default value for Code25MinOccurrenceLength changed from 5 to 7.

The definition for the DIB functions (e.g ScanBarCodeFromDIB) have been updated so the DIB handle is defined as HANDLE instead of long.

23.25 Version 8.4.1.1

The structure of the installation folders has been changed to make it easier to understand which DLL files form the core part of the toolkit.

Interfaces for .Net Core and Python have been added.

The QRCodeAutoMedianFilter property in the .Net interface has been changed from bool to short. The property is in fact a mask and should not have previously been designated bool in this interface.

Improvement to reading rectangular datamatrix barcodes.

23.26 Version 8.3.3.8

Fixed floating point exception in Windows XP systems when decoding DataMatrix barcodes.

Fixed potential crash problem if a system or process runs out of memory.

The leading byte in a Kanji byte pair must now be in the range 0x80 to 0x9F unless QRCodeByteMode has been set to a value of 6 (Kanji only mode).

The software is now built using ZLib version 2.11.

23.27 Version 8.3.3.6

SoftekBarcodeCOM converted from multi-byte to Unicode

SoftekBarcodeNet now works consistently with Unicode characters on both Unicode and Ansi systems.

Auto detection of UTF-8, Kanji, Big5 and GBK improved for Qr Codes with the default for QRCodeByteMode changed from 1 to 2 to better conform with the ISO specification for QRCode. Big5 and GBK characters will only be interpreted when QRCodeByteMode is set to the correct value.

23.28 Version 8.3.3.5

Fix for random barcode coordinates when MultipleRead is not enabled.

Fix for false positive EAN-13 Supplemental barcodes (when a UPC-E barcode is adjacent to an EAN-13 barcode and also at the edge of an image).

Fixed small memory leak when detecting Code 25 barcodes.

Improvements to PDF-417 recognition:

The wide bars in guard patterns can now be processed even when they are split.
Left hand guard patterns with smaller than normal wide bars can now be processed.

A new property called EdgeThreshold has been added. This improves performance of large color images, especially for DataMatrix recognition. To retain 100% compatibility with previous versions this property should be set to 0.

The general speed of Data Matrix recognition has been improved.

Support for dot pattern Data Matrix has been improved with the addition of the new property called DotDataMatrixSupport.

23.29 Version 8.3.3.3

Performance improvement for images withcertain types of repeated pattern.

Fix for recognition of Kanji characters using the Kanji byte compaction scheme and addition of the QRCodeByteMode property to handle the ambiguity inherent in the QRCode specification and optional support for the Big5 and Gbk character sets through the QRCodeByteMode property.

Fixed potential memory overrun when the ReportUnreadBarcodes mask contains 1.

Fix for some Codabar false positive readings.

Added method called GetRawBarStringBytes to SoftekBarcodeNet.dll

Resolved issue with incorrect coordinates of Codabar barcodes.

The Debenu PDF Library has been updated from version 1611 to 1613.

Improvements to PDF417 detection:

Error correction has been improved where the data includes padding code words.

Barcodes with an extra vertical bar before the final large bar can now be decoded using default settings.

The calculation of the number of columns has been improved.

23.30 Version 8.3.3.2

Added new flag to PdfImageRasterOptiuons: 64 means only use Debenu direct access.

Improvement to recognition of black and white skewed QrCodes.

Added new advanced property called Show2DCornersInResults.

Fixed bug that could make recognition of some PDF417 barcodes inconsistent.

23.31 Version 8.3.3.1

The Foxit/Debenu Quick **PDF Library** has been upgraded to version 1611

Improvements have been made to **Code 93** recognition.

The handling of the **Datamatrix** upper shift character has been fixed.

If the DataBarOptions mask includes the value 256 then the full Databar type for the bar code is returned by GetBarStringType.

Improvements to the handling of EAN13 Supplemental barcodes.

False positive results for **Code 25** bar codes in some documents containing tables with borders have been fixed.

An issue with loading certain PDF documents on Windows 10 systems has been fixed.

23.32 Version 8.3.2.1

PDF Processing

The Foxit/Debenu Quick PDF Library has been upgraded to version 1511.

If the toolkit is unable to open a document using the Foxit/Debenu direct access functions then it will automatically revert to the normal functions for processing a document.

The DLL files for the different render engines supported are now included in the bin folder by default.

23.33 Version 8.3.1.1

Skewed Barcode Handling

Backward compatibility warning:

This version of the toolkit handles skewed document differently to previous versions, with the introduction of the DeskewMode property. At its default value of 2 the SkewTolerance property no

longer has any effect. If you wish to handle skewed documents as in previous versions of the toolkit then please set DeskewMode to 0.

Data Matrix Recognition

2 new properties added for data matrix recognition:

DataMatrixRectangleSupport - can be used to process extended rectangular formats or to ignore all rectangular formats (and get a performance gain).

DataMatrixSearchLevel – balance speed against read rate.

PDF Handling

This version can handle password protected PDF documents through the use of the PdfPassword property.

The toolkit has been upgraded to use version 14 of the Debenu library.

Other changes…

The prototypes for both stScanBarCodeFromBitmap and mtScanBarCodeFromBitmap have changed so that the handle for the bitmap is passed as type HBITMAP rather than long.

23.34 Version 8.2.1.4

Ugraded Debenu DLL version to 13.12.

Code 25 recognition improvements:

- Fixed potential memory leak.
- Added new advanced property called Code25PitchVariation.
- Increased general speed recognition

Fixed BEX error in Code 128 recognition.

Fixed memory leak in PDF417 recognition.

Improved QR-Code recognition on multi-processor systems.

Improved speed of recognition from JPG files on multi-processor systems.

23.35 Version 8.2.1.1

Introduction of a qualiy score for barcodes that indicates how clear the barcode was in the image.

Update for Debenu PDF library to version 13, which means that the toolkit now supports PDFium.

Default for PdfLocking changed from 1 to 2

The following Advanced properties have now been added as properties to most interfaces of the toolkit:

Code128DebugMode
Code128Lenient
Code128SearchLevel
Code39MaxRatioPcnt
DataMatrixAutoUTF8
DataMatrixFinderGapTolerance
MedianFilterBias
PDF417MacroEscapeBackslash
PatchCodeMinOccurrence
Pdf417AutoUTF8
Pdf417ChannelMode
Pdf417Debug
PdfLocking
QRCodeAutoMedianFilter
QRCodeAutoUTF8
QrCodeReadInverted
QuotedPrintableCharSet
RotateBy45IfNoBarcode
UseOldCode128Algorithm
UseRunCache
WeightLongerBarcodes

23.36 Version 8.1.2.8

Performance improvement in Code 128 recognition. New advanced property called Code128SearchLevel added.

QR-Code recognition - Quiet zones now enforced correctly in black and white images, giving a performance increase.

Improvement to function that compares points in barcodes with those of existing barcodes, giving a performance improvement.

Added a new advanced property called PdfLocking that controls access to PDF documents.

Added a new advanced property called DataMatrixParams to allow changes to the recognition of data matrix timer patterns.

Improved recognition of PDF417 barcodes that have a large variance in column width.

23.37 Version 8.1.2.7

SoftekBarcodeNet.dll has reverted to .Net 3.5 and the .Net 4 version placed in a folder called dotnet4.

SoftekBarcodeLib.dll and SoftekBarcode64Lib.dll have both reverted to .Net 2 and the .Net 4 versions placed in a folder called dotnet4.

The default value for the MultipleRead property has been changed from false to true. Applications that do not set the MultipleRead property in their settings should be altered to explicitly set this property to false.

The default value for the MaxThreads property has been changed from 0 to 4.

23.38 Version 8.1.2.6

Fixed recognition of 2 column compact PDF-417 barcodes.

Added new advanced property called Code128DebugMode.

23.39 Version 8.1.2.5

Updated Debenu library files to version 12.12

23.40 Version 8.1.2.3

Improvement to Code 25 recognition for module sizes of 2 or less.

The Debenu PDF library has been updated to version 12.

23.41 Version 8.1.2.1

SoftekBarcodeNet.dll, SoftekBarcodeLib.dll and SoftekBarcode64Lib.dll are now built against the .Net framework 4.0.

New property called DataMatrixFinderGapTolerance to control the size of gaps allowed in the L shaped finder pattern of a data matrix bar code.

Improvements to error correction in data matrix bar codes of size 16X16 or smaller.

Inverted QrCodes (i.e. white on black) are now recognized. This feature can be turned off with the property QrCodeReadInverted.

SoftekBarcodeLib.dll and SoftekBarcode64Lib.dll now both handle Unicode file paths correctly for PDF files.

Improved handling of PNG documents.

Added the advanced property called RotateBy45IfNoBarcode.

Improved the detection of large skewed data matrix bar codes.

23.42 Version 8.1.1.10

Increased tolerance of variation in column width for PDF-417 bar codes.

Updated Debenu PDF DLL files to version 11.15

Added new advanced property called Code39MaxRatioPcnt.

Changed Code 39 recognition to reduce the number of false positive reads and correctly read some bar codes where the wide white bars are the same size or smaller than the narrow black bars or vice versa.

23.43 Version 8.1.1.9

Fixed error in GS1-databar alogorithm that prevented recognition of certain stacked bar codes.

Fixed multi-threading error in decoding evaluation licenses. Note that this error does not apply to full license keys.

23.44 Version 8.1.1.6

Added new advanced properties called PDF417AutoUTF8 and DataMatrixAutoUTF8. These new properties prevent the toolkit from double encoding data that is already encoded using UTF-8. The documentation has also been updated to include QRCodeAutoUTF8.

Modified PDF-417 recognition to accept guard patterns that are missing the 4th bar on the right hand side.

23.45 Version 8.1.1.5

Added work around for issue where Debenu library returns PNG file labelled as BMP.

Fixed exception for PDF files containing a single black and white BMP image.

23.46 Version 8.1.1.4

Removed calls to WaitForSingleObject during FreeLibrary in SoftekBarcodeDLL.dll.

Added new property called Code128Lenient.

23.47 Version 8.1.1.3

1. Changed function specifications in the .Net wrapper interface (SoftekBarcodeNet.dll) from Cdecl to Stdcall.
2. Median filter capability added for color images.

23.48 Version 8.1.1.1

23.48.1 Main changes:

1. All the DLL files now come in a single folder and most of the names for DLL's have been changed.
2. New functions have been added in the DLL, COM and .Net (SoftekBarcodeNet.dll) interfaces to support background reading of bar codes.
3. The SDK can now take advantage of multi-core systems. Pages in multi-page documents can be processed in parallel and with a single page the horizontal and vertical scanning can take place in parallel.
4. The PDF Extension has been completely redesigned with native x86 and 64-bit conversion provided by Debenu. Split PDF files will now retain the original format. It is still possible to use the old render tool (VeryPDF) on x86 systems via the PdfImageRasterOptions setting.

23.48.2 Background bar code reading

Version 8 of the SDK has a new set of function that allows applications to start background scans of documents. The scans may be monitored for progress and terminated at any time during the scan.

ScanBarCodeInbackground - launch a scan for a bar code in the background

ScanBarCodeWait - wait for a background scan to complete for a number of milliseconds

GetProgress - get %age progress of a background read.

GetBarCodeCount - get number of bar codes found so far in a background read

ScanBarCodeAbort - abort a background read

GetScanExitCode - get the exit code from a background read

23.48.3 Faster processing of documents

Version 8 of the SDK can now process separate pages and parts of images in separate threads. This can give much improved speeds for multi-page documents and single page documents containing 2-D bar codes. The maximum number of threads is controlled by the new property called MaxThreads.

23.48.4 File name changes:

Information on file name changes can be found here

23.48.5 Class Name Changes

Information on class name changes can be found here

23.48.6 PDF Extension

Information on changes to the PDF Extension can be found here

23.48.7 No longer supported

Information on files and interfaces that are no longer supported can be found here

23.48.8 Changes to SDK properties

PdfImageRasterOptions is no longer used by the toolkit and defaults to 0. Applications should set it to 0 for future compatibility.

PdfImageExtractOptions is no longer used by the toolkit and defaults to 0. Applications should set it to 0 for future compatibility.

MaxThreads a new (advanced) property that controls how many internal threads an instance of the SDK may use.

23.49 Version 7.6.1.4

Added the following advanced flags:

- MedianFilterBias
- Pdf417MacroEscapeBackslash
- QRCodeBWAutoMedianFilter

Fixed problem that prevented recognition of Code 93 bar codes.

Improved datamatrix algorithm.

Change to Code 128 algorithm that allows for a larger final bar.

23.50 Version 7.6.1.1

The main changes for version 7.6.1 are in the areas of performance control and recognition of skewed bar codes.

Performance Control

TimeOut is a new property that specifies the maximum amount of time the toolkit should spend searching for a barcode within a single page of a document. It does not include the time taken to load a page from disk into memory. The property defaults to 5000 (ms) and so could potentially cause different behaviour to previous versions. To remove the time out the property to be set to a value of 0.

MaxBarcodesPerPage is also a new property and specifies the maximum number of barcodes expected in a single page of an image. This has the potential to dramatically improve the speed of processing because the toolkit will stop processing a page as soon as a bar code is discovered. Note that this property only has effect if MultipleRead is set to True. The default is 0 which means no limit to the number of barcodes on a single page.

There are 2 other new properties in this area:

When UseFastScan is True and only 1 bar code is expected on a page, the toolkit will perform an initial high speed scan of an image. If it cannot find a bar code using the fast scan then the normal slower scan will take place.

BarcodesAtTopOfPage can improve performance if MultipleRead is False or MaxBarcodesPerPage is non-zero. When set to True the toolkit will start the search at the top of a page.

Skewed Barcodes

Version 7.6.1 can handle certain types of skewed bar code differently to previous versions and the following properties may remove the need to use the SkewTolerance property for certain bar codes.

When the SkewedLinear property is True (default) the toolkit will decode many skewed 1-D bar codes that previous versions would only decode with with the use of the SkewTolerance property. The SkewedDatamatrix property works in a similar way.

Reporting of Unread Barcodes

This version of the toolkit also has some limited capabilities to report bar codes that could not otherwise be detected.

ReportUnreadBarcodes is a mask that controls the reporting of barcodes that could not otherwise be successfully decoded. Please note the following limitations:

- Maximum of 1 unread barcode reported per page
- No unread barcodes reported on a page where another barcode was successfully read.
- Linear barcodes must consist of at least 40 black bars.
- 2-D barcodes will only be reported if they reach and fail the error correction phase of decoding.
- Any large series of parallel lines will be reported as an unread linear barcode
- All unread barcodes reported as type “UNREAD” with value “UNREAD”
- The appropriate 2-D type must be enabled to detect unread barcodes of that type.
- If a linear barcode type is disabled then barcodes of that type may be reported as unread.

Other changes

- Support for <PropertyName/> as an alternative to <PropertyName></PropertyName>
- Auto detection of UTF8 data via QRCodeAutoUTF8 advanced property.
- Can now handle QRCodes containing ECI data. ECI value of 26 is now recognized as an indicator that data is encoded using UTF8 format.
- FNC1 characters are now handled correctly in datamatrix barcodes.
- Support for mirrored QR-Codes.
- SoftekSDKDemo command line options added.
- Added function mtGetRawBarString for applications that need to handle binary data.
- Added new advanced property called Pdf417ChannelMode.

23.51 Version 7.5.1.35

- Correction to the decoding of Base 256 encoded datamatrix barcodes.
- Support added for QR-Code symbols starting with FNC1 characters.
- Increased max size of QR-Code from 2K to 4K and improved detection for version 0 QR-Codes.

- Improvements to Code 128 and Code 39 barcode reading. The SDK now handles single vertical lines to the left or right of symbols and within the quiet zone.
- Corrected possible memory access problem on 64-bit systems when opening TIF documents.

23.52 Version 7.5.1.29

QR-Codes:

- Fix for symbols containing more than 500 characters.
- Improvements to recognition of multiple QR-Codes in the same image and at different angles.
- Improvement to recognition of skewed QR-Codes.
- Improved support for Version 40 symbols.
- Fix for a divide by zero error for certain QR-Codes.
- Improved support for symbols with badly proportioned finder targets.
- GetBarStringDirection now returns the correct value for QR-Codes.

Datamatrix:

- Improved support for symbols with parallel lines close by.
- Bug fixes for correct decoding EDIFACT encoded data.
- Bug fixes for correct decoding of German characters.

PDF-417 error correction improvements (the SDK was using one more error correction codeword than was necessary). This makes the most noticeable improvement when using the smallest numbers of error correction code words.

Bug fixes for Splitting TIF documents when not including the pages containing barcodes.

23.53 Version 7.5.1.22

If the height of a bitmap (HDIB or HBITMAP) is negative then the SDK will perform a vertical flip on the image before processing.

Improved support for rotated QR-Codes.

Black and white png files are now loaded with the correct photometric.

Improved support for splitting TIF files compressed using “old style” jpeg compression (type 6). Gray scale is now supported but full color is not supported.

Refined fix from version 7.5.1.18 (if TifSplitPath is “%s.tif” or “%s.pdf” and no barcode is detected in an image then the output file will not be created) to include values such as c:\temp\%s.tif. i.e If the base name evaluates as empty then no output file will be created.

Support added for upper shift characters in datamatrix barcodes.

Improved read rates for PDF-417 barcodes.

23.54 Version 7.5.1.18

PDF Extension

The SDK is now able to extract images from specific pages in a PDF document. This means that an image only PDF document can be processed one page at a time. In the case where only a single barcode be recognized or a single page processed, thre will be a considerable improvement in speed. Options have been added to PdfImageExtractOptions to control this single-pass or multi-pass behavior. Note that additional dll files are also supplied for the PDF extension with this version (cximagecrt.dll and tiffcp.dll).

Fix for inaccurate bouding rectangles for certain adjacent barcodes.

If TifSplitPath is "%s.tif" or "%s.pdf" and no barcode is detected in an image then the output file will not be created.

Improvements to Datamatrix recognition, including the prevention of false positive results.

Fix for loading bmp files on 64-bit systems.

23.55 Version 7.5.1.12

- Improvement to the handling of PDF-417 barcodes with more than 10 columns of data.
- Run-caching enabled which will give better performance for some images containing 2-D barcodes.

23.56 Version 7.5.1.10

- Improvement to datamatrix recognition algorithm.
- Allowance of greater variation in the width of UPC guard patterns.
- SoftekSDKDemo "run tests" output format fixed.
- Potential memory over-run fixed.

23.57 Version 7.5.1.6

- Bug fix for the decoding of QR-Code data stored using numeric compaction.
- Support for structured append QR-Codes added to the toolkit.
- Support for Kanji characters in QR-Codes
- Bug fix for the decoding of PDF417 barcodes using error level 0.
- Bug fix for BMP files with non-standard offsets to the start of the image data.
- Improvement to the Code 128 recognition algorithm.
- Default character set encoding changed from raw to UTF-8.

23.58 Version 7.5.1.1

Addition of support for QR-Codes.

23.59 Version 7.4.2.1

Enhanced error information

New functions have been added to the toolkit to retrieve error codes that may assist with the detection and correction of problems. The GetlastError function returns the last internal error code for the toolkit and GetLastWinError returns the last windows error number.

Suppression of Dialog Messages

The SDK will now suppress all dialog boxes under its control when the process is not visible (e.g. when being run as part of a service). This will prevent processes from hanging.

Partial reads on PDF-417 barcodes

It was possible for certain PDF-417 barcodes to give partial and incorrect readings. This has been corrected. The maximum length for a PDF-417 barcode has now been increased from 2K to 8K (when represented in quoted printable format).

2-D barcodes at coarse fax resolution

Previous versions of the toolkit were not able to decode 2-D barcodes from coarse resolution faxes because the vertical and horizontal resolutions were different. This has been corrected.

Improvements to the SoftekSDKDemo application

The SoftekSDKDemo application will now retain settings in the registry. It is also possible to reset the settings back to default and import and export settings from/to an xml file.

23.60 Version 7.4.2.2

64-bit version of SoftekBarcodeLib2.dll now correctly packaged as a strongly named assembly.

Potential buffer over-run in the Code-128 module fixed.

23.61 Version 7.4.2.3

Support for Desktop specific licensing added to the toolkit.

23.62 Version 7.4.1

23.62.1 Version 7.4.1.1

1. License Key

The most important change in Version 7.4.1 is the requirement to set a license key prior to calling the ScanBarCode, ScanBarCodeFromBitmap or ScanBarCodeFromDIB functions. This will simplify the distribution of updates to the toolkit and protect the rights of licensees.

2. New .net component

This version also includes a new interface to the toolkit for .net applications. The SoftekBarcodeLib3.dll component wraps around the SoftekBarcode.dll and can process TIF documents at 3 times the speed of the SoftekBarcodeLib2.dll component.

3. Simplified installation

All the files for the SDK are now installed into a single folder by running a self extracting installer. No dll files need to be registered unless the COM or OCX interfaces are to be used and so the new version may be used in isolation to previous versions of the toolkit. The install set also includes the dll files for the PDF Extension, with the license key determining whether or not barcode can be read from PDF documents.

4. PDF Processing

Image only PDF documents can now be processed at up to 3X the speed of previous versions. When the PdfImageOnly property is set to True (the default value) the toolkit will assume that all PDF documents are image only (e.g, scanned documents) and use a faster conversion method than would be needed for other types of PDF document.

5. PDF Processing on x64 Systems

This version has resolved a number of issues with processing PDF documents on x64 based systems.

6. Replacement of ImageReader and DeveloperCenter with SoftekSDKDemo

The demonstration applications used in previous versions of the toolkit have been replaced with the SoftekSDKDemo application, which comes in versions for both x86 and x64 based systems.

7. Improvements to barcode recognition

Improvements have been made to the Code 39, Code 128, GS1-Databar and Datamatrix modules.

8. Code 93 Support

Support for Code 93 barcodes has also been added. Please refer to the manual page for further details.

9. TIF Resolution

If a TIF document uses a resolution of 1 dpi then the value will be ignored and a default value of 200 dpi used instead.

10. DLL Dependencies

The only DLL file that requires installation of the Microsoft Visual C++ Redistributable Package (2008 SP1) is SoftekbarcodeLib2.dll.

23.62.2 Version 7.4.1.2

A PInvoke error when using the SoftekBarcodeLib3.dll under .Net 4 has been fixed.

23.62.3 Version 7.4.1.3

The SetScanRect method in SoftekBarcodeLib3.dll now functions correctly.

Options for extracting images from PDF documents can now be controlled through the PdfImageExtractOptions advanced property.

A typing error in the product names for the redistribution packages has been corrected.

23.62.4 Version 7.4.1.4

The LicenseKey property added to Java class

An error in the PDF-417 decode algorithm has been corrected.

The datamatrix module has been changed to better handle the effects of perspective in a photograph.

A correction has been made to the way BMP files are loaded by the win32 dll.

23.62.5 Version 7.4.1.5

PdfImageRasterOptions mask property added to allow control over rasterization of PDF documents.

Options dialog in SoftekSDKDemo has been split into tabs.

23.62.6 Version 7.4.1.6

Updated versions of pdf2tif.dll, convert.dll and encrypt.dll added to the install set

23.62.7 Version 7.4.1.7

Added FilePathEncoding property.

Support for file paths containing non-ascii characters has been improved.

23.63 Version 7.3.1

1. GS-1 Databar

GS-1 Databar support has been added to toolkit. This includes RSS-14, RSS-14 Stacked, RSS Truncated, RSS Limited, RSS Expanded and RSS Expanded Stacked. There is also support for supplemental information encoded in micro-PDF-417 barcodes.

2. Color Images

A new property called ColorProcessLevel has been added to the toolkit. This property controls the amount of time the toolkit will spend processing a color image and should make it unnecessary to set the ColorThreshold property to a value other than zero. If you are upgrading from a previous version of the toolkit and set ColorThreshold to a non-zero value in your code then you will need to change the value to 0 in order to take advantage of the new property.

3. Micro-PDF-417

A memory violation error has been fixed in the micro PDF-417 module.

4. Size of color PDF files

The compression used when outputting PDF documents as part of the Tiff Split process has been changed from zip to jpeg, which yields significantly smaller files.

5. Java Interface

The properties; ReadMicroPDF417, ReadDatabar and ReadDataMatrix have been added to the java interface.

6. SkewTolerance Range

Range checks have been added for the value of the SkewTolerance property.

7. SoftekBarcodeLib2.ScanBarCodeFromBitmap empty string problem

It was possible for the ScanBarCodeFromBitmap method in the .net 2 interface to return a value of 1, and yet GetBarString would return an empty string. This has been fixed and the barcode value is now returned.

8. Median Filter memory leak

A memory leak has been fixed when MedianFiler is set to True.

9. Double read on skewed DataMatrix barcodes.

An error whereby it was possible to get a double read for slightly skewed datamatrix barcodes has been fixed.

10. Empty values for Datamatrix

An error whereby the toolkit returned empty datamatrix barcode values for some images has been fixed.

23.64 Version 7.2.1

1. Datamatrix Barcode Support

Support has been added for Datamatrix ECC 200 barcodes. To enable the reading of datamatrix barcodes please set the ReadDataMatrix property to True.

2. Memory leak fix

Previous versions of the toolkit contained a memory leak if SkewTolerance > 0 and UseOverSampling was set to True or 1.

3. Support for 2 and 3 digit Code 25 barcodes.

If MinLength is set to an appropriate value then the toolkit will recognize 2 and 3 digit Code-25 barcodes.

4. Potential memory over-run when reading micro-PDF417 barcodes fixed.

5. Output of split files in PDF Format

A potential memory exception has been fixed when export TIF split pages in PDF format.

6. Reading non-TIF files from URL's

A bug which meant that the toolkit always assumed a file read from a URL was in TIF format has been fixed.

23.65 Version 7.1.4

1. Tiff-split in PDF Format

The tiff-split feature of the toolkit is now able to output in both tiff and pdf format. Note that the .Net component requires installation of the PDF Extension to do this where as the DLL, OCX and

COM interfaces do not. If the PDF Extension is installed then the toolkit is also able to split a PDF file into smaller parts. Note that PDF files are rasterized before being split.

2. Java Interface Extended

The Java interface has been extended to include tiff-split functionality.

3. PdfDpi default value changed

The default value of the PdfDpi property (the resolution at which PDF files are rasterized) has been changed from 200 to 300.

4. Patch Code Recognition

Previous versions of the toolkit would output false positive results if ReadPatchCodes was set to True and MinOccurrence and PrefOccurrence were left at default values, but version 7.1.4 imposes a minimum hit count for patch codes of 30. This value can only be changed via the LoadXMLSettings method/function by setting the property PatchCodeMinOccurrence.

5. PDF-417 Recognition

Improvements have been made to PDF-417 recognition, which allow the toolkit to cope with larger variations in the width of columns.

23.66 Version 7.1.3

1. Improved support for the EAN-13 family of barcodes.

The EAN-13 module has been improved to allow a greater variance in module width.

2. Extended support for Code 2 of 5 barcodes.

Support added for the following variants of the Code 25 family: Datalogic, Matrix,

Industrial and IATA via the ReadCode25ni setting.

3. Improved detection of false positive results when using SkewSettings > 0.

In previous versions it was possible to get partial reads of barcode such as Code 2 of 5 when using SkewSetting values > 0. This version detects and removes such errors.

4. Encoding property added to control format in which barcodes are returned:

0 (default) = raw (with null characters suppressed)

1 = Quoted printable

2 = Unicode

3 = UTF-8

Note that this property mainly applies to PDF-417 barcodes.

5. Sorting of barcode results

Previous versions of the toolkit have generally returned barcodes ordered bottom up and left to right - with some exceptions. This version now checks the ordering to ensure that barcodes are returned in the expected order.

6. Bug fix in SaveResults method in the .net 2.0 component.

In previous versions this function only saved the results for the last page in a multi-page TIF document.

7. The properties PdfBpp, PdfDpi, PageNo and Photometric can be exported and imported using the XML interface.

8. Added support for micro-PDF-417 barcodes via the ReadMicroPDF417 property.

9. Added support for Short Code 128 barcodes via the ReadShortCode128 property. Another property called ShortCode128MinLength has also been added to control the minimum allowed length for this type of barcode. The default value is 2.

10. Improved recognition of long Code 128 and Code 39 barcodes.

23.67 Version 7.1.2b

1. EAN-13/UPC-A Barcodes

Correction to potential parity checking problem.

23.68 Version 7.1.2

1. Code 39 Module

The Code 39 module has been improved to handle barcodes with split bars.

2. Java Interface

SetScanRect, GetBarStringRect and GetBarStringPage methods added to the jni interface. Please see the README.txt file in the java folder for further details.

3. Tiff Split

The TiffSplitPath property can now include %s as well as %d. The %s code will be replaced by the value of the barcode in the sub-file.

4. JPG Loading Errors

The handling of error conditions when loading a jpg file has been improved.

23.69 Version 7.1.0a

Modified DeveloperCentre application to automatically switch to XML settings if no barcode found.

23.70 Version 7.1.0

Median Filter, XML and Java Interfaces.....

1. Median Filter

The Median Filter option is another method of cleaning noisy images. It's more versatile than the noise reduction filters because it isn't restricted to a single orientation, although it isn't recomended for low resolution images.

2. XML Interface Added

2.1 Using an XML File to Control Properties

It is now possible to load settings for the toolkit from an XML file. What's more, it is also possible to define groups of settings, to be applied successively to an image until a barcode is found. Groups of settings can also be targetted at particular pages in an image.

See the manual pages for LoadXMLSettings and ExportXMLSettings for more details.

2.2 Exporting Results to an XML or CSV File

The SaveResults method can be used to export barcode values and locations to an XML or CSV file.

See the manual page for SaveResults for more details.

2.3 Using an XML File to Specify Files or Folders to be Processed.

An XML file can also be used to spedify files and folders for the toolkit to scan, with the results written to another XML file or a CSV file.

Note that this does not currently support PDF files.

See the manual page for ProcessXML for more details.

3. Java Interface

The toolkit is now provided with a Java class, which can be found in the java folder under the installation folder. Please see the README file in the Java folder for more details on the supported properties and methods.

4. Code 25 character width tolerance

The allowed variation in width for Code 25 characters in barcodes of more than 6 characters has been increased.

5. Extra wide spaces

The toolkit now handles barcodes with an exceptionally high ratio between the first black bar and first white space.

6. Auto Color Threshold Calculation Improved

The automatic calculation of the value for color threshold has been imrpoved.

7. PDF-417 Recognition

An improvement has been made to the PDF-417 algorithm, which allows it to recognise much lower resolution images than was previously possible. A logic error has also been fixed in the decoding alogorithm for byte compacted data.

8. Code 39 Recognition

A small improvement has been made to the Code 39 algorithm to handle barcodes with a low ratio between the wide and narrow bars.

23.71 Version 7.0.10

1. Code 128 recognition.

A small change to help read barcodes where the size of the black bars have been shrunk by the scanning process.

23.72 Version 7.0.9

1. Auto-calculation of Color Threshold

The algorithm to to calculate the color threshold value has been improved to handle a greater variety of color images. The new method also recognizes color images that only use 2 unique colors (i.e. black and white encoded as color).

2. Code 128 recognition

The variation in character width for a code 128 barcode has been increased to allow better detection in color images. A restriction in character width variation is still used for reasons of performance.

3. BMP files using less than 24 bpp and with no image size specified in the header.

If a BMP file wirh less than 24 bits per pixel contains no image size in the header then the toolkit will now calculate the expected image size from the width and height.

4. Memory leak when Using Noise Reduction on Color Images

If a color bitmap was processed with the NoiseReduction property set to a value great than zero and ScanDirection exclusively portrait or landscape, then a memory leak occurred. The leak has now been fixed.

23.73 Version 7.0.8

1. A logic error in TifSplitMode has been fixed.

If TifSPlitMode was set to zero then the second barcode in the image would appear on the last page of the first file, rather than the first page of the second file.

23.74 Version 7.0.7

1. Code 128 recognition

The variation in character width for a code 128 barcode has been increased.

23.75 Version 7.0.6

1. Photometric property added to the Managed Components

23.76 Version 7.0.5

1. The following combination of settings in versions 7.0.3 and 7.0.4 was causing the ScanBarCode function to return a value of 0, but has been corrected in this version.

PageNo != 0
MultipleRead = True
ScanDirection = 1, 4 or 5

2. The recognition of PDF-417 Barcode with small column widths has been improved.

3. The default value for the ReadPDF417 property has been changed from TRUE to FALSE

23.77 Version 7.0.4

1. Automatic setting of ColorThreshold

If the ColorThreshold property is set to a value of zero then the toolkit will automatically calculate a value. This enables the toolkit to read barcodes from very dark or very light images.

2. Code 128 Recognition

The tolerance level for differences in the width of Code 128 characters has been increased.

23.78 Version 7.0.3

1. PNG Image Support

The Windows DLL, OCX and COM interfaces now all support the PNG file format.

2. PDF-417 Recognition Improved

The PDF-417 module has been improved to handle images with the following defects:

a. The scanning process has reduced the size of the black bars.

b. Part of the end pattern is missing.

c. The column widths vary through the image.

d. The bases of some columns are missing.

3. Code 39 Error Correction/Oversample Bug

A buffer over-run which occurred on some images where Code 39 error correction and oversampling had both been set has been fixed.

23.79 Version 7.0.2

1. Splitting TIF files according to the location of the barcodes in the image.

A new mode has been added, which will split a TIF file and throw away the pages containing a barcode. For example, if an image consists of 10 pages, with barcodes on pages 1, 4 and 6 then the toolkit will create 3 new files. The first file will contain pages 2 and 3. The second will contain page 5 and the third will contain pages 7, 8, 9 and 10.

Set TifSplitMode to a value of 2 to use this mode.

2. Code 128 recognition improved.

During the scanning process it is possible for the black bars to increase in size and for the white spaces to shrink. The toolkit has always allowed for this problem, but it meant that the Code 128 module sometimes mis-read a character value and so failed to return a value for the barcode. This has been corrected and some barcodes that were not detected will now read OK.

23.80 Version 7.0.1a

1. PDF-417

PDF-417 reading capability has been added to the toolkit (please see note above). Control over PDF-417 reading is via the ReadPDF417 property, which by default is set to False.

2. Code 25 Module modified to handle differences in character widths in high resolution images.

3. MaxLength increased from 99 to 999.

4. Code 39 checksum calculation fixed.

23.81 Version 6.2.1a

1. Version numbers brought into line on the dll and ocx files.

2. Code 25 module - maximum ratio value increased from 3 to 4.

23.82 Version 6.2.1

1. Code 39, Code 25 and Codabar Recognition

The rules for the above barcode symbologies have been tightened to reduce the possibility of a false positive reading.

23.83 Version 6.2.0d

1. Code 39 Recognition

Further enhancements have been made to Code 39 recognition:

If the Code39Checksum property is set to True then the toolkit will only report Code 39 barcodes where the last character is a valid checksum for the rest of the string.

If the ExtendedCode39 property is set to True then the toolkit will attempt to interpret the barcode in the Code 39 Extended symbol set.

2. Code 25 Recognition

The Code 25 recognition module has been modified to allow for barcodes where the width of a module for a bar may be different to that of a space.

3. Regular Expression Matching

The Pattern property now allows applications to specify a regular expression that all reported barcodes must match against. For example, Pattern = "^ABC[0-9]+XYZ$" would match barcodes similar to "ABC123456XYZ".

If the related property called Numeric is set to True then it is the equivalent of setting Pattern to "^[0-9]+$".

4. Improved Code 128 Recognition

The Code 128 recognition module has been improved to cope with barcodes where the bar widths have been distorted by the process of scanning or faxing.

5. Error Correction

If the ErrorCorrection property is set to True then the toolkit will attempt to find the best match for a barcode character where it is practical to do so. This currently only applies to Code 39 barcodes, but will be extended to other types in the future.

6. Minimum Space Bar Width

The MinSpaceBarWidth property specifies the smallest width of a space in a barcode. Any spaces smaller than this width will be ignored by the toolkit. This is very useful when a scanned image contains lots of white dots in the black bars. Setting the property to a value of 2 or 3 can often result in a reliable read for such images.

7. Control over PDF Conversion

Where the PDF Extension to the toolkit is installed, it is now possible to control the resolution and color depth of the image created from the PDF file, and used to locate the barcodes. This allows applications to control the speed and accuracy of the process. The PdfBpp property controls the number of bits-per-pixel of the converted image (1, 8 or 24) and PdfDpi controls the number of dots per inch of the converted image. Lower values for either property lead to an increase in speed.

8. Default value of LineJump property has been changed from 9 to 1.

9. Improvement in speed

Optimization in the Code 25, Code 39 and Code 128 module has given a significant improvement in speed.

23.84 Version 6.1.1

1. PDF Conversion

The ScanBarCode method now uses the PDF Extension to convert the pdf document at 8 bits per pixel.

2. ZIP Compression

The toolkit now supports TIF documents that use ZIP compression.

3. ScanDirection Mask Change

The managed component interface has been modified to interpret the ScanDirection mask in the same way as the standard windows dll. This may mean that some applications will need changing if they use the managed component and set a non-default value for the mask. If in doubt then please contact support@bardecode.com for further information.

4. Percentage mapping Mode for SetScanRect

A new mapping mode is now available for SetScanRect. If a value of 1 ise used for the mapping mode then the units are treated as a percentage of the width or height of the image.

5. SoftekATL COM Object Trial Version

The trial version of the SoftelATL COM Object has been modified to prevent the display of the trial version dialog box, which caused problems for web servers. The new trial version replaces the last 3 characters of the barcode with a * character.

6. HBITMAP and DIB Support in the Managed Component

The managed component interface now supports the ScanBarCodeFromBitmap and ScanBarCodeFromDIB methods.

7. UPC-E Recognition

A bug that prevented the recognition of valid UPC-E barcodes has been fixed.

8. Images with Small Height

The toolkit now correctly handles images with a height of less than 4 pixels.

9. Old JPEG Compression

If a TIF file contains pages that use old jpeg compression then the return value will either be the number of barcodes found on other pages or, if no other barcodes were found, a value of -5.

10. Code 39 and Code 25 Recognition

The algorithms for Code 39 and Code 25 have been improved to cope better with poor quality barcodes.

11. Memory Leak in the .Net Component

A memory leak has been fixed in the .Net component. The memory was leaked each time an instance of the component was destroyed.

23.85 Version 6.1.0

1. PDF File Format

When used in conjunction with the "SoftekBarcode Toolkit PDF Extension" library, this version of the toolkit can read barcodes from PDF files.

2. Improvement in Performance

The time is takes for the toolkit to load TIF files has been significantly improved by code optimization. This means that barcodes can be read from TIF files at much higher speeds than before (does not apply to the managed component).

3. Code 25 Non-Interleaved Recognition available across all interfaces

The ReadCode25ni property has now been added to all interfaces. The default for this property is False.

4. Photometric Property Added

The Photometric property is used with the ScanBarcodeFromBitmap method to specify the bit value representing Black in a 2 color bitmap. Default of 0.

5. AllowDuplicateValues Property Added

The AllowDuplicateValues property controls whether the toolkit allows barcodes with identical values on the same page to be ignored. Default of True

6. Properties of Managed Component not used when calling ScanBarCodeFromBitmap method.

Any properties set in the managed component were previously being ignored when calling the ScanBarCodeFromBitmap method. The method has now been modified to fix the settings each time it is called.

7. Softek COM Object Memory Problem Fixed

A problem in the COM object interface to the toolkit has now been fixed. This caused the BSTR values returned from certain functions to become corrupt.

8. Full Multi-Page Support for Managed Component

The Managed Component now provides full multi-page image support. The default for the PageNo property is now zero - which means that all page of an image will be checked for barcodes.

23.86 Version 6.0.10

1. GIF file support for DLL, OCX and COM Interfaces.

This also includes support for multi-page GIF files.

2. Multi-page Image Support in the Managed Component Interface

The .Net managed Component now supports barcode reading from multi-page image files. It still differs from the regular DLL though because it will only search the page in the image specified by the PageNo property. If PageNo is set to zero then the component will throw an exception when the image is loaded.

3. Access Violation Problems with OverSampling

When OverSampling was used with certain images the barcode toolkit read 1 scan line beyond the length of the image and caused an access violation. This has now been corrected.

4. ColorThreshold level problem resolved

A problem with setting high values for ColorThreshold has been resolved.

23.87 Version 6.0.9

1. The ScanBarCodeFromBitmap function has been fixed so that the HBITMAP handle isn't deleted.

2. Problem Reading from 32 bits-per-pixel bitmap handles.

The function that converts 32-bit color images to black and white used an incorrect value for the color threshold, which meant that perfectly good barcodes were not detected by the software.

3. The ScanBarCodeFromBitmap function has been changed to read the bitmap in the same direction as the toolkit reads other image formats. This means that any rectangle specified by the SetScanRect method should now be measured in pixels from the top left hand corner, rather than the bottom left hand corner.

23.88 Version 6.0.8

1. Memory leak fixed in ScanBarCodeFromBitmap

23.89 Version 6.0.7

1. Access Violation in ScanBarCodeFromBitmap

A small section of code that tried to determine if a DIB handle had been passed to ScanBarCodeFromBitmap has been removed because it was causing access violations. DIB handles should be passed to the ScanBarCodeFromDIB function.

2. The 100% managed component is now "strongly named".

23.90 Version 6.0.6

1. Support for JPEG compressed TIF files.

2. Bug fixed in the splitting of multi-page tif files.

Description: It was possible for barcodes in a multi-page TIF file to be repeated. This has been fixed.

3. libbarcode.jpg file is no longer required.

Description: The functionality for reading JPG files has been moved inside the SoftekBarcode.dll file.

4. Non-Interleaved Code-25 Capability

The functions stSetReadCode25ni and stGetReadCode25no have been added to the DLL interface. These functions have the same calling convention as stSetReadCode25 and stGetReadCode25 and control whether the toolkit searches for non-interleaved Code 25 barcodes. The default is for the toolkit not to search for these barcodes.

23.91 Version 6.0.4

1. Exception Error in Dot Net Component

Description: Certain image files caused an exception error in the dot net component, especially when using options such as noise reduction or over sampling.

Solution: SoftekBarcodeLib.dll has been modified to interpret the byte width of image scan lines correctly.

2. Reading Barcodes from DIB Handles

A new function called ReadBarCodeFromDIB has been added to the toolkit. The old function called ReadBarCodeFromBitmap will now try to determine if the handle is for a BITMAP or a DIB and call the ReadBarCodeFromDIB function if necessary.

3. Reading Skewed Barcodes.

The 45 degree angle mask values for the ScanDirection mask are no longer used. The reading of skewed barcodes is now controlled by the SkewTolerance property which ranges in value from 0 (no check for skewed barcodes) to 5 (all angles considered). The default value is 0.

23.92 Version 6.0.3

1. Code 39 barcodes with length less than 4 characters would not read.

Solution: The toolkit will now read Code 39 barcodes of any length. Remember though that the purpose of the MinLength setting is to cut down on the chance of a false positive reading.

2. Photometric Interpretation of Monochrome BMP files - when using the .Net component.

The corresponding fix from version 6.0.2 has been applied to the .Net Component.

23.93 Version 6.0.2

1. Photometric Interpretation of Monochrome BMP files

Description: BMP files that used a zero bit to represent a white pixel were being displayed in negative by the image viewer and interpretted as a negative image by the barcode reader.

Solution: SoftekBarcode.dll has been modified to read the color map correctly.

2. Some BMP files would read OK on Windows 2000 but would fail to read on Windows XP.

Description: The bmWidthBytes member of the BITMAP structure is sometimes mis-reported by Windows XP. The value is often rounded up to an integer divisible by 4. This leads to the barcode toolkit loading the image into memory incorrectly and either missing the barcode, getting the poisition wrong or reporting multiple occurrences of a barcode.

Solution: The length of the bitmap data is now checked to make sure that it matches the reported size of bmWidthBytes. The fix is in SoftekBarcode.dll.

The photometric interpretation on monochrome bitmap files is now checked before processing. Files that represented white with a zero bit were being read as a negative image

23.94 Version 6.0.1

The main change to the barcode toolkit for version 6 is the ability to work in a multi-threaded environment. A managed component has been added to the set of interfaces ,and allows easy deployment of the toolkit within the .Net framework.

The algorithms for the Code 3 of 9 and Code 2 of 5 Interleaved barcode types have been re-written to provide a better success rate with low resolution images.

It's also worth noting why version 5 was never released. Just as version 5 got to the pre-release stage, the need for a thread safe version of the toolkit was identified and was given top priority.

Summary of changes:

1. Support for the Codabar symbology.

The toolkit now supports recognition of the Codabar barcode type. The ReadCodabar property has been added, with a default value of True.

2. Improved recognition algorithms for Code 3 of 9 and Code 2 of 5 Interleaved.

Both of the algorithms for detecting Code 3 of 9 and Code 2 of 5 Interleaved barcodes have been improved. This particularly applies to low resolution images where there may only be minor differences between the wide and narrow bars.

3. 100% Managed .Net Component added to the range of interfaces.

The managed component is implemented by the file SoftekBarcodeLib.dll. This interface supports most of the properties and methods of the toolkit with the exception of the following:

ScanBarCodeFromBitmap

TifSplitPath

TifSplitMode

BitmapResolution

The component uses the .Net class for loading images and so supports a greater range of image types (e.g PNG, GIF) in addition to the usual set.

4. UPC-E to EAN-13 Conversion

UPC-E is a zero suppressed version of UPC-A/EAN-13 and the default in version 4 was to restore the suppressed zero's. The option has been added in version 6 to leave the barcode as it is printed, via the ConvertUPCEToEAN13 property. The default value is True - which means that the zero's will be restored (the behaviour in version 4).

5. Retrieve the Direction of a Barcode in an Image

A new method has been added to retrieve the orientation of a barcode. The GetBarStringDirection method returns a value that, if used for the ScanDirection property, would detect the barcode. Some barcodes can be read from left to right or right to left. In these cases the direction returned is the first match found.

6. Code39Draw control

The Code39Draw control has been fixed so that it prints out the same size as it displays on the screen.

7. 45 degree angle scanning

4 new scan directions are available in the toolkit. The default value for the ScanDirection mask is still set to 15, which covers left-right and top-down directions. Please refer to the manual for full details of the new values for the mask.