Morovia DataMatrix Font & Encoder 5 Reference Manual

Morovia DataMatrix Font & Encoder 5 Reference Manual

All contents of this document are furnished for informational use only and are subject to change without notice and do not represent a commitment on the part of Morovia Corporation or its subsidiaries (Morovia). Reasonable effort is made to ensure the accuracy of the information contained in the document. However, Morovia does not warrant the accuracy of this information and cannot accept responsibility for errors, inaccuracies or omissions that may be contained in this document.

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH MOROVIA® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN A SIGNED AGREEMENT BETWEEN YOU AND MOROVIA, MOROVIA ASSUMES NO LIABILITY WHATSOEVER, AND MOROVIA DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF MOROVIA PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY INTELLECTUAL PROPERTY RIGHT OF A THIRD PARTY.

Publication date: May 2017
Revision: 9902

Technical Support
Phone: (905) 752-0226
Fax: (905) 752-0355
Email: support@morovia.com
Web: http://www.morovia.com

For more information about Morovia products, visit http://www.morovia.com.

Table of Contents

1. Introduction 1
  1.1. What's New 1
  1.2. Backward compatibility 2
  1.3. Installing DataMatrix Font & Encoder 2
    1.3.1. To Install From a CD 2
    1.3.2. To Install from Direct Download 2
  1.4. Limitations of Trial Version 2
2. Overview 3
  2.1. What is Data Matrix? 3
  2.2. Working with DataMatrix Font & Encoder 5 4
    2.2.1. Font-Based 5
    2.2.2. Image File Based 6
    2.2.3. ActiveX Control 6
3. Using DataMatrix Encoder GUI 7
  3.1. Exporting Images 8
  3.2. Program Options 9
4. Encoder DLL API Reference 11
  4.1. ImageTypeEnum 11
  4.2. DataMatrixEncode 12
  4.3. DataMatrixEncode2 13
  4.4. DataMatrixEncode2W 13
  4.5. DataMatrixResultGetSizeID 14
  4.6. DataMatrixResultGetBarcodeString 14
  4.7. DestroyDataMatrixEncodeResult 15
  4.8. PaintDataMatrixImageRaster 15
  4.9. PaintDataMatrixImageVector 16
  4.10. PaintDataMatrixImageRaster2 17
  4.11. PaintDataMatrixImageVector2 18
  4.12. PaintDataMatrixImageEMF 18
  4.13. DataMatrixGetErrorMessage 19
  4.14. DataMatrixResultGetBarcodeString2 19
  4.15. PaintDataMatrixImageClipboard 20
5. DataMatrix ActiveX Control Reference 21
  5.1. Specification 21
  5.2. Properties 21
  5.3. BackColor, ForeColor Properties 22
  5.4. ModuleWidth Property 22
  5.5. Picture Property 22
  5.6. SizeID Property 23
  5.7. TargetDPI Property 23
  5.8. Text Property 23
  5.9. CopyToClipboard Method 24
  5.10. ExportImageRaster Method 24
  5.11. ExportImageVector Method 24
  5.12. GetActualSizeID Method 25
6. Adding DataMatrix to Crystal Reports 27
  6.1. UFL Functions 27

6.2. Tutorial: Step by Step 28
6.3. Distributing UFL, Fonts with your report application 30
7. Adding DataMatrix to Microsoft Office Documents 31
7.1. Access (ActiveX Control) 31
7.2. Access (using DataMatrix Font) 32
7.3. Excel (ActiveX Control) 32
7.4. Microsoft Word 34
7.5. Word Mail Merge 34
8. Adding DataMatrix Symbols to SQL Server Reporting Service 37
8.1. Custom Assembly 37
8.1.1. Installing Custom Assembly 37
8.2. Adding DataMatrix Barcodes to Report 38
8.3. Deployment 39
A. Technical Support 41
B. Input format (ECI and Structural Append) 43
B.1. Tilde codes 43
B.2. Macro 5 and 6 44
B.3. Extended Channel Interpretation (ECI) 44
B.4. Structural Append (SA) 44
B.4.1. File ID (FID) 45
B.4.2. Sequence Indicator (SI) 45
B.4.3. Total Number of Symbols 45
C. Size ID Parameter (updated in version 5.1) 47
C.1. Modifier byte 47
C.2. Size ID byte 47
C.3. Royal Mail MailMark Barcode 48
D. Unicode String Encode Support 49
E. Fontware License Agreement 51
Glossary 53
Index 55

List of Figures

2.1. Example Data Matrix Symbol 3
3.1. DataMatrix Encoder Dialog Options 7
3.2. Export Options Dialog 9

List of Tables

2.1. Data Matrix Sizes 3
2.2. DataMatrix Font Characteristics 5
4.1. List of Enumerations 11
4.2. List of Functions 11
4.3. ImageTypeEnum Enumeration 12
5.1. DataMatrix Control Specification 21
5.2. List of DataMatrix Control Properties 21
5.3. List of DataMatrix Control Methods 21
6.1. List of Crystal Reports UFL Functions (`cruflDataMatrix5.dll`) 27
B.1. Macro 5 and 6 44
C.1. Size ID value for MailMark barcodes 48

Chapter 1. Introduction

DataMatrix Font & Encoder 5.0 is the ultimate tool box to print DataMatrix symbols. Data matrix is a variable-size, two-dimensional symbology capable of encoding large amount of data. The barcode consists of an array of data cells within a distinct perimeter pattern. Although it is capable of encoding 2000 characters, data matrix is best for encoding small amount of data when space is a primary concern. Data matrix is a space efficient symbology. When encoding small amount of data, it produces more compact barcodes than many other two dimensional symbologies.

Both fonts and encoder functions are updated in this release, based on the feedback received from customers using previous version. Both 32-bit and 64-bit encoder DLLs and applications are included in the package. Windows 2000 or above is required to run run encoder GUI, or to call the encoder DLL. True type fonts can be used in other platforms such as OS/X or Linux.

This package includes the following contents:

- One true type font targeting laser printers and bar code printers - `mrvdatamatrix5.ttf`.
- Scalable PCL font to be used on PCL-compatible printers.
- DataMatrix Encoder GUI, a GUI program to create barcode strings based on data entered. The program can export barcode images in a variety of formats, such as *PNG*, *EMF*, *SVG* and *EPS*.
- A Windows native *DLL* that allows you to add DataMatrix printing to your own application.
- A Crystal Reports extension DLL that adds DataMatrix Code printing functionality to Crystal Reports 9.0 and above. Both 32-bit and 64-bit Crystal Runtimes are supported.
- An ActiveX Control that can be inserted into Microsoft Office programs or integrated into your custom application.
- Examples demonstrating how to add DataMatrix printing functionality to your applications in a variety of programming environments, such as Access, Word, Excel, and .Net.

1.1. What's New

Version 5.1

- Added support on fine tuning encoder behavior. Allow user to specify the encodation mode selection method; and allow user to specify whether the encoder increases the size automatically if the size specified is too small. The support is implemented by enhancing existing size ID parameter.
- Added support to encode Unicode strings into a datamatrix barcode.

Version 5.0

- New font design. Some software calculate line heights incorrectly and causes gaps between two rows. In this release, the font is redesigned to work around such problems.
- Image creating support. This package allows your application to create image files directly, without the presence of the font.
- complete 64-bit support. This release aims seamless integration with 64-bit applications, not just "work on 64-bit Windows" as our competitors claimed. All DLLs and Executables have both 32-bit and 64-bit versions. For example, 64-bit Office 2010 is supported.
- ActiveX Control. This package includes an ActiveX control that you can insert directly into Microsoft Office programs, such as Word and Excel. You can also use the control in your custom applications.
- 64-bit Crystal Reports support is added. Previous version can only be deployed on a 32-bit Crystal Report runtime.

- Updated VBA module to support 64-bit Microsoft Office products.

1.2. Backward compatibility

In version 5.x, the data matrix font is updated to make it work more consistently among a large number of applications. In order to achieve this, the font is redesigned and no longer remains backward compatible Whit early 3.x versions. You can't use the version 5 fonts in conjunction with an encoder from the early version, and vice versa. We changed font names and encoder file names to reflect this change. Keep in mind that some knowledge base articles, and forum discussions are for earlier versions. They are likely not applicable on the new version.

1.3. Installing DataMatrix Font & Encoder

In order to provide seamless application experience, the installer is greatly enhanced. The setup actually comprise two installers - the 32-bit version that installs 32-bit only files, and 64-bit version that installs both 32-bit and 64-bit components. Therefore, in any supported Windows system you always get the right files installed.

1.3.1. To Install From a CD

1. Insert the program CD into your CD drive. The setup starts automatically. Or if the auto-run feature isn't enabled on your system, click the Windows Start button and choose the Run command. Type **`D:\Setup.exe`** in the dialog box and click the OK button (Note that `D` represents the letter assigned to your CD-ROM drive. If your drive is assigned to a different letter, use it instead of `D`).
2. Follow the on-screen instructions.
3. Your will be prompted to enter the `License To` and `Registration Code`. The `License To` and `Registration Code` information are found on the back of the CD case.

1.3.2. To Install from Direct Download

1. Click the Download link to start the download.
2. When the browser prompts, do one of the following: A. To run setup immediately, click `Open` or `Run This Program from Its Current Location`. B. If you decide to run the setup at a later time, click `Save` or `Save This Program to Disk`.
3. If you chose `Save This Program to Disk` in Step 2, locate the file where you saved it, and double click on it.
4. Follow the setup instructions.
5. Your will be prompted to enter the `License To`/ `Registration Code`. The `License To` and `Registration Code` information can be found in the email we send to you after order completes.

1.4. Limitations of Trial Version

A trial version is provided on our web site that can be downloaded freely. In trial version, barcode created will have additional text "DEMO" encoded at the end. The full version does not have the limitation.

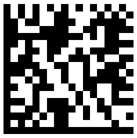
Chapter 2. Overview

This chapter briefly reviews the symbology (a.k.a. barcode formats) supported by DataMatrix Fonts & Encoder 5 - Data Matrix.

2.1. What is Data Matrix?

Data matrix is a variable-size, two-dimensional symbology capable of encoding large amount of data. The barcode consists of an array of data cells within a distinct perimeter pattern. Although it is capable of encoding 2000 characters, data matrix is best for encoding small amount of data when space is a primary concern. Data matrix is a space efficient symbology. When encoding small amount of data, it produces more compact barcodes than any other two dimensional symbologies.

Figure 2.1. Example Data Matrix Symbol



The encoding process (converting data into an array of light/dark modules) is fairly complicated. First, to produce space efficient symbols, data matrix standard defines six character sets, also called *encodation modes* with each targeting a different kind of data. Data matrix encoder shifts the encodation mode from one to another whenever it sees that doing so will reduce the whole symbol size. Moreover, to make sure that the symbol can be read when some portions become damaged, the encoder adds additional modules for error correction purposes. The error correction algorithm by itself is not easy.

There are two types of data matrix bar codes: ECC 000-140 and ECC 200. The difference lies in the error correction algorithms they employ - the ECC 000-140 uses several levels of convolutional error correction and ECC 200 uses Reed-Solomon error correction. By employing Reed-Solomon error correction algorithm, ECC200 allows data to be successfully read, even if 60% of the data area is damaged. The most recent data matrix standard requires all new applications to use ECC 200.

Data matrix offers limited combinations between the number of rows and columns. A valid selection is called a “data matrix size”. All data matrix sizes are listed in the table below. A data matrix symbol can be any one of the 30 shapes.

Table 2.1. Data Matrix Sizes

Size ID		**Symbol Size**		**Data Capacity**		
Enum	**Value**	**Row**	**Column**	**Numeric**	**Alphanum**	**Binary**
mbxDMTargetSize_10X10	0	10	10	6	3	1
mbxDMTargetSize_12X12	1	12	12	10	6	3
mbxDMTargetSize_14X14	2	14	14	16	10	6
mbxDMTargetSize_16X16	3	16	16	24	16	10
mbxDMTargetSize_18X18	4	18	18	36	25	16
mbxDMTargetSize_20X20	5	20	20	44	31	20

Size ID		Symbol Size		Data Capacity		
Enum	**Value**	**Row**	**Column**	**Numeric**	**Alphanum**	**Binary**
mbxDMTargetSize_22X22	6	22	22	60	43	28
mbxDMTargetSize_24X24	7	24	24	72	52	34
mbxDMTargetSize_26X26	8	26	26	88	64	42
mbxDMTargetSize_32X32	9	32	32	124	91	60
mbxDMTargetSize_36X36	10	36	36	172	127	84
mbxDMTargetSize_40X40	11	40	40	228	169	112
mbxDMTargetSize_44X44	12	44	44	288	214	142
mbxDMTargetSize_48X48	13	48	48	348	259	172
mbxDMTargetSize_52X52	14	52	52	408	304	202
mbxDMTargetSize_64X64	15	64	64	560	418	278
mbxDMTargetSize_72X72	16	72	72	736	550	366
mbxDMTargetSize_80X80	17	80	80	912	682	454
mbxDMTargetSize_88X88	18	88	88	1152	862	574
mbxDMTargetSize_96X96	19	96	96	1392	1042	694
mbxDMTargetSize_104X104	20	104	104	1632	1222	814
mbxDMTargetSize_120X120	21	120	120	2100	1573	1048
mbxDMTargetSize_132X132	22	132	132	2608	1954	1302
mbxDMTargetSize_144X144	23	144	144	3116	2335	1556
mbxDMTargetSize_8X18	24	8	18	10	6	3
mbxDMTargetSize_8X32	25	8	32	20	13	8
mbxDMTargetSize_12X26	26	12	26	32	22	14
mbxDMTargetSize_12X36	27	12	36	44	31	20
mbxDMTargetSize_16X36	28	16	36	64	46	30
mbxDMTargetSize_16X48	29	16	48	98	72	47

The size ID parameter used by our software is an interger. If value 0-29 are passed, it indicates the encoder should create a barcode that matches the size. However, it may increase the size if the encoder finds that the size requested is too small.

In version 5.0, two additional values are supported: -1 and -2. If -1 is specified, the program selects the minimum size that meets the requirement, and searching from the rectangular shape (0~23). If -2 is specified, the program searches from the square shape (24) first. And if no square shape sizes can encode the data, it then searches all rectangular shapes.

In version 5.1 and above, the descrition above still applies. However, size ID parameter is further enhanced to allow user to specify encodation mode selection and whether the encoder should increase the size if the size is determined too small. For details, refer to Appendix C, *Size ID Parameter (updated in version 5.1)*.

2.2. Working with DataMatrix Font & Encoder 5

The methods to create barcodes are greatly expanded in this version. You should select one based on your working environment - Font, image or ActiveX control.

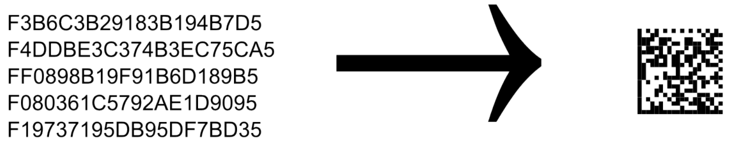
2.2.1. Font-Based

Font-based approach creates a datamatrix barcode by formatting the encoder output with data matrix font.

> **Note** In almost all circumstances, you can not just type your number and format with a barcode font to create a valid barcode. You must generate the barcode string first, and format the barcode string with the font.

The Font-based approach is easy to understand, and in some occasions the only way to add barcode printing functionality, such as in Crystal Reports.

The drawing below illustrates how you create a DataMatrix symbol using font-based approach. You first create the barcode string (which is an array of text lines). Format the barcode string with `MRV DataMatrix5` font you get the barcode.



```
F3B6C3B29183B194B7D5
F4DDBE3C374B3EC75CA5
FF0898B19F91B6D189B5
F080361C5792AE1D9095
F19737195DB95DF7BD35
```

The font characteristics are listed as below:

Table 2.2. DataMatrix Font Characteristics

Filename	Typeface	Module With (under 6 points)
mrvdatamatrix5.ttf	MRV DataMatrix5	20 mils (0.5mm)

2.2.1.1. Encoding

To create a valid barcode, you need to call an encoder to get a special string, and format this string with our DataMatrix font. To get this string, you need to call an encoder - you can run DataMatrix Encoder GUI, and obtain the result from this GUI program. Or if you need to bulk generate the results, using the programming interface exposed from the encoder DLL.

DataMatrix Encoder GUI

This GUI program allows you to quickly create barcode string and transfer it to other programs. You are able to enter the data encoded, and create the barcode on the fly. For more information, see Chapter 3, *Using DataMatrix Encoder GUI*.

MoroviaDataMatrixFontEncoder5.dll

This DLL is a standard Windows DLL that can be called by many programming environments, including Microsoft Office, C++, FoxPro and .Net. Most programming environments support Windows DLL directly. We provide a VBA module and a C# class that wrap around the DLL functionality.

uflDataMatrix5.dll

This DLL is specifically designed for Crystal Reports. For more information on using the software with Crystal Reports, see Chapter 6, *Adding DataMatrix to Crystal Reports*.

2.2.2. Image File Based

You can export barcodes as standard image files to be consumed in other word processing and imaging programs. The DataMatrix Encoder GUI supports exporting images in four formats via a GUI interface. The Encoder DLL and DataMatrix ActiveX Control also provides programming interfaces to export images in those formats:

- PNG (Portable Network Graphics)
- BMP (Windows Bitmap File)
- EMF (Ehanced Meta File)
- SVG (Scalable Vector Graphics)
- EPS (Encapsulated PostScript)

Special attention should be paid when the target printer has has low resolution, such as a thermal printer or fax machine. When generating vector graphics files, the software allows you to specify target dpi which you should set to the one that matches your printer. It is generally not recommended to scale the image down or up when working with low-resolution printers.

2.2.3. ActiveX Control

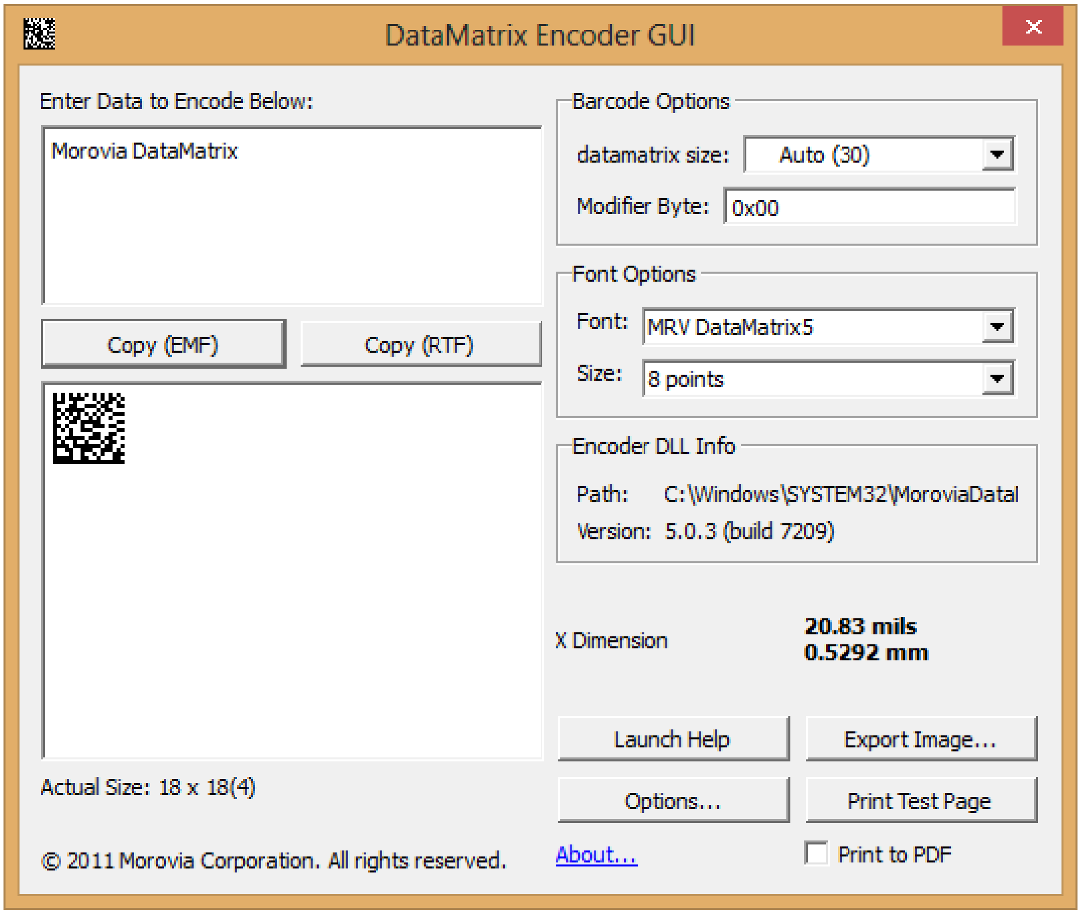
Many environments support ActiveX Controls, such as Microsoft Word, IE and Excel. You can insert the DataMatrix ActiveX Control into documents, spreadsheets, or invokes the control at the background in your program. Unlike standard image formats which does not contain encoding parameters, such as size and data encoded, ActiveX control keeps those info. Therefore you can change the data matrix by editing the properties.

Using ActiveX Control in Microsoft Office programs is straightforward. Programming knowledge is required in order to integrate the control with your custom programs.

Chapter 3. Using DataMatrix Encoder GUI

DataMatrix Encoder GUI provides a fast way to create DataMatrix barcodes on the fly. From the program you can easily create barcode strings or images and transfer them to a graphics designer or word processor.

Figure 3.1. DataMatrix Encoder Dialog Options



DataMatrix Encoder GUI

Enter Data to Encode Below:

Morovia DataMatrix

Barcode Options

datamatrix size: Auto (30)

Modifier Byte: 0x00

Font Options

Font: MRV DataMatrix5

Size: 8 points

Encoder DLL Info

Path: C:\Windows\SYSTEM32\MoroviaDataI

Version: 5.0.3 (build 7209)

X Dimension 20.83 mils 0.5292 mm

Actual Size: 18 x 18(4)

© 2011 Morovia Corporation. All rights reserved.

□ Print to PDF

Using this GUI program is straightforward. Enter the data encode in the edit box. The barcode changes as you type. You can change the barcode size by changing the font size. The following lists the major items in the dialog.

Data to Be Encoded

This is the place to enter the data to be encoded. Multiple-line text is supported. To enter the next line, enter **Ctrl+Enter**.

Copy (EMF)

Place the barcode image in the clipboard, so that you can subsequently paste it into another application such as Microsoft Word. The EMF created does not use font, so that you do not need to copy the font file when you view document from another computer.

Note that X dimension in the resulted barcode is determined by the Module Size in the Options dialog, as the EMF format does not use the font.

Copy (RTF)

Place the barcode string in the clipboard, so that you can subsequently paste them into another application such as Microsoft Word.

Both rich text format (RTF) and text format are placed in clipboard. In applications that are capable of processing *RTF* (Rich Text Format), you will see a barcode immediately after pressing the paste button. Otherwise, you will see the text string instead. If this is the case, highlight the whole string and format with “MRV DataMatrix5” font.

Barcode Info

This section displays the actual size of the bacode. In the dialog above, the actual barcode has a size ID of 5 (20 by 20 modules).

Launch Help

View the program help.

Options

Pops up the Options dialog where you can specify additional option for exporting barcode images. See Section 3.2, “Program Options” for more information.

Export Image

Click on this button to export the barcode into standard image formats such as EMF, SVG, EPS and PNG.

About

Click on this button to pop up the About dialog. License info is accessible through this link.

X Dimension

This section displays the X-dimension of the barcode, in units of mils and mm.

Encoder DLL Info

This section displays the absolute path and the file version of the encoder DLL. This is useful to troubleshooting potential DLL confilicts.

Font Options

Select the font and size from this section.

Barcode Options

Specify the size desired. The size is divided into two parts: the left siginiicant byte which indicates the datamatrix size, and a modifier byte that can be used to adjust encoder behavior. The modifier byte is specified through a hexdecimal value. See Appendix C, *Size ID Parameter (updated in version 5.1)* for more information.

3.1. Exporting Images

Datamatrix Encoder GUI supports exporting into several standard image formats. The images generated do not contain references to the fonts so that you can move around them without requiring the software to be installed.

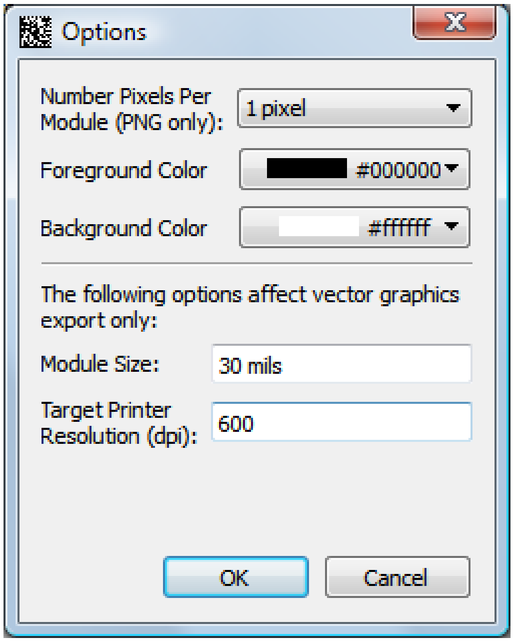
- *EMF*. This is the default vector graphics format used on Windows operating system.
- *PNG*. PNG is a bitmap image format widely supported by imaging software. It is the recommended image format for web graphics.
- *BMP*. BMP is the default bitmap format used on Windows platform.
- *EPS*. This format is widely used in graphics design industry. Recommended to use in conjunction with Adobe software such as Adobe Illustrator.

- *SVG*. SVG is an emerging vector graphics format. FireFox 3 supports it out of the box, as well as many drawing programs such as Adobe Illustrator.

3.2. Program Options

You can specify a couple of options applicable in exporting barcode images.

Figure 3.2. Export Options Dialog



Number Pixels Per Module (PNG only): 1 pixel

Foreground Color: #000000

Background Color: #ffffff

The following options affect vector graphics export only:

Module Size: 30 mils

Target Printer Resolution (dpi): 600

Number Pixels Per Module

This option applies on raster images only. Specify the number of pixels of a module (the real estate unit of a PDF417 barcode).

Foreground Color / Background Color

Specify the color for image background and foreground. Color values are specified using six hexidecimal digits, with the components in the order of red, green and blue. For exmaple 000000 is the value for black color, and ff0000 is the value for red. Note that changing color may reduce the readablity of the barcode.

Module Size

Apply on Vector image format only. You can specify a nominal size for the module width. You can enter something like "20 mils", or "0.02mm". If unit of measure is not specified, the unit of mils is assumed.

Target Printer Resolution

Apply on vector image format only. If you export a vector graphic images to a low resolution device such as screen or thermal printer, you should fill this field with the value of the resolution. For screen, use 96. This value makes sure that the length units are properly aligned to the edge of pixels when rasterized.

Chapter 4. Encoder DLL API Reference

Creating DataMatrix barcodes using DLL APIs involves three steps. The first step is to call `DataMatrixEncode` or `DataMatrixEncode2` with data to encode and size option. If this step succeeds, a pointer is returned from which you can query the attributes of the barcode created, "paint" the barcode into an image file, or obtain the barcode string. Finally, you should call `DestroyDataMatrixEncodeResult` to release the memory resource used by the encoder result object.

Table 4.1. List of Enumerations

Enumeration	Comment
ImageTypeEnum	Image formats supported by the encoder.

Table 4.2. List of Functions

Function	Comment
DataMatrixEncode	Encodes data and returns a pointers that points to encoder result object.
DataMatrixEncode2	Abbreviated version of DataMatrixEncode.
DataMatrixEncode2W	The function **DataMatrixEncode2W** is added in 5.1 release to encode Unicode strings.
DataMatrixResultGetSizeID	Retrieves the size ID of the datamatrix barcode created.
DataMatrixResultGetBarcodeString	Retrieves the barcode string that becomes a DataMatrix symbol after being formated with a DataMatrix font.
DestroyDataMatrixEncodeResult	Releases all resources allocated to the encoder result object.
PaintDataMatrixImageRaster	Writes the DataMatrix barcode in raster image format specified to a disk file.
PaintDataMatrixImageVector	Writes the DataMatrix barcode in vector image format specified to a disk file.
PaintDataMatrixImageRaster2	Writes the DataMatrix barcode in raster image format specified to IStream object.
PaintDataMatrixImageVector2	Writes the DataMatrix barcode in vector image format specified to an IStream object.
PaintDataMatrixImageEMF	Creates a Windows Enhanced Meta File object and returns the handle.
DataMatrixGetErrorMessage	Retrieves a string that describes the error.
DataMatrixResultGetBarcodeString2	Returns barcode string without requiring user to preallocate the buffer.
PaintDataMatrixImageClipboard	Creates a Windows Enhanced Meta File object and place it into the clipboard.

4.1. ImageTypeEnum

Image formats supported by the encoder.

```
enum ImageTypeEnum {
    IMAGE_PNG== 0,
    IMAGE_SVG== 1,
    IMAGE_EMF== 2,
    IMAGE_EPS== 3,
    IMAGE_BMP== 4
};
```

Remarks

The ImageTypeEnum enum has the following values:

Table 4.3. ImageTypeEnum Enumeration

Constant	Value	Comment
IMAGE_PNG	= 0	PNG (Portable Network Graphics)
IMAGE_SVG	= 1	SVG (Scalable Vector Graphics)
IMAGE_EMF	= 2	EMF (Windows Enhanced MetaFile)
IMAGE_EPS	= 3	EPS (Encapsulated PostScript)
IMAGE_BMP	= 4	BMP (Windows Bitmap)

PNG and BMP are raster formats, which store color information of individual pixels. SVG, EMF and EPS are vector graphics formats and contains drawing commands instead. If you are using raster graphic format, one pixel should be mapped to one or integral times pixels on the printer. If you are using vector graphics format, the drawing units should map to integral times pixels on the printer. They are referenced in the PaintDataMatrixImageRaster function and PaintDataMatrixImageVector function.

4.2. DataMatrixEncode

The **DataMatrixEncode** function encodes data and returns a pointers that points to encoder result object.

```
int __stdcall  DataMatrixEncode(
    const char * dataToEncode,
    int sizeIDRequested,
    void ** ppResult
);
```

Parameters

dataToEncode

[in] Pointer to a null-terminated string containing the data to be encoded. You can use tilde codes to encode control characters such as NUL, as well as advanced features such as ECI and structural append. For more information, see Appendix B, *Input format (ECI and Structural Append)*.

sizeIDRequested

[in] An integer value that corresponds to the size ID of the datamatrix barcode. Since version 5.1, this parameter is enhanced to support additional parameters, such as encodation mode selection. See the manual for details.

ppResult

A pointer to a pointer that points to the encode result created. Use this pointer to discover the actual attributes of the symbol created, or create image files.

Return Values

If the function succeeded, the return value is 0. If the function failed, it returns the error code. You can call function DataMatrixGetErrorMessage to obtain the description of the error.

Remarks

This function encodes the data according to the parameter specified, and returns a pointer from which you can retrieve the encoding result.

4.3. DataMatrixEncode2

The function **DataMatrixEncode2** is an abbreviated version of DataMatrixEncode.

```
void *__stdcall    DataMatrixEncode2(
    const char * dataToEncode,
    int sizeIDRequested
);
```

Parameters

dataToEncode

[in] Pointer to a null-terminated string containing the data to be encoded. You can use tilde codes to encode control characters such as NUL, as well as advanced features such as ECI. For more information, see Appendix B, *Input format (ECI and Structural Append)*.

sizeIDRequested

[in] An integer value that corresponds to the size ID of the datamatrix barcode. Since version 5.1, this parameter is enhanced to support additional parameters, such as encodation mode selection. See the manual for details.

ppResult

A pointer to a pointer that points to the encode result created. Use this pointer to discover the actual attributes of the symbol created, or create image files.

Return Values

If the function failed, the return value is 0 (NULL). If it succeeds, it returns a pointer that points to the result object. You should release the encoder result object by calling function DestroyDataMatrixEncodeResult.

Remarks

In this function, a pointer is returned instead of status code. A non-null return value indicates that no error occured. The pointer associated resource must be released by calling DestroyDataMatrixEncodeResult.

The function **DataMatrixEncode2** is an abbreviated version of DataMatrixEncode.

4.4. DataMatrixEncode2W

The function **DataMatrixEncode2W** is added in 5.1 release to encode Unicode strings.

```
void *__stdcall    DataMatrixEncode2W(
    const wchar_t * dataToEncode,
    int sizeIDRequested
);
```

Parameters

dataToEncode

[in] Pointer to a null-terminated string containing the data to be encoded. You can use tilde codes to encode control characters such as NUL, as well as advanced features such as ECI. For more information, see Appendix B, *Input format (ECI and Structural Append)*.

sizeIDRequested

[in] An integer value that corresponds to the size ID of the datamatrix barcode. Since version 5.1, this parameter is enhanced to support additional parameters, such as encodation mode selection. See the manual for details.

ppResult

A pointer to a pointer that points to the encode result created. Use this pointer to discover the actual attributes of the symbol created, or create image files.

Return Values

If the function failed, the return value is 0 (NULL). If it succeeds, it returns a pointer that points to the result object. You should release the encoder result object by calling function DestroyDataMatrixEncodeResult.

Remarks

See the manual on the details how a wide string is handled. In this function, a pointer is returned instead of status code. A non-null return value indicates that no error occured. The pointer associated resource must be released by calling DestroyDataMatrixEncodeResult.

4.5. DataMatrixResultGetSizeID

The **DataMatrixResultGetSizeID** function retrieves the size ID of the datamatrix barcode created.

```
int __stdcall  DataMatrixResultGetSizeID(
    void * pEncodeResult
);
```

Parameters

pEncodeResult

[in] The pointer that points to the encoding result object, returned from DataMatrixEncode or DataMatrixEncode2 functions.

Return Values

The size ID of the DataMatrix barcode.

4.6. DataMatrixResultGetBarcodeString

The **DataMatrixResultGetBarcodeString** function retrieves the barcode string that becomes a DataMatrix symbol after being formated with a DataMatrix font.

```
int __stdcall  DataMatrixResultGetBarcodeString(
    void * pEncodeResult,
    char * buffer,
    unsigned int * maxSize,
    const char * eol
);
```

Parameters

pEncodeResult

[in] The pointer that points to the encoding result object, returned from DataMatrixEncode or DataMatrixEncode2 functions.

buffer

[in] The pointer that points to a byte array that receives the string.

maxSize

[in,out] Pointer to a variable that specifies the size of the buffer pointed to by the buffer parameter, in bytes. When the function returns, this variable contains the size of the data copied to buffer.

eol

[in] Pointer to a NUL terminated string that will be appended to each row in the barcode string.

Return Values

If the function succeeded, it returns the length of the string (excluding terminating NUL character). If the funtion fails due to insufficient storage, it returns 0.

Remarks

When creating barcodes using font-based solution, you first call encoder function DataMatrixEncode or DataMatrixEncode2 to obtain a pointer that points to the result object. Then DataMatrixResultGetBarcodeString is called to obtain a string that becomes DataMatrix symbol after the font is applied. If memory is not an issue, allocate a large buffer with 8096 bytes to hold the largest DataMatrix barcode. This size is sufficient for the largest symbol with carriage return and line feed as line ending.

4.7. DestroyDataMatrixEncodeResult

The **DestroyDataMatrixEncodeResult** function releases all resources allocated to the encoder result object.

```
void __stdcall DestroyDataMatrixEncodeResult(
    void * pResult
);
```

Parameters

pResult

[in] Pointer to the encoder result object.

Remarks

After the object is destroyed, the specific pointer pResult is no longer valid.

4.8. PaintDataMatrixImageRaster

The **PaintDataMatrixImageRaster** function writes the DataMatrix barcode in raster image format specified to a disk file.

```
int __stdcall  PaintDataMatrixImageRaster(
    void * pEncodeResult,
    const wchar_t * wszFilename,
    int pixelsPerModule,
    int forecolor,
    int backcolor,
        int imageType
    );
```

Parameters

pEncodeResult

[in] Pointer that points to the encoder result object.

wszFilename

[in] Pointer to the file name for the image file to be created.

pixelsPerModule

[in] Number pixels per module with.

forecolor

[in] Value of the foreground color, in RGB colorspace.

backcolor

[in] Value of the background color, in RGB colorspace.

imageType

[in] An integer that indicates the image file format. The current version supports two formats: PNG(0) and BMP(4).

Return Values

If the function failed, it returns the error code. You can call function DataMatrixGetErrorMessage to obtain the description of the error.

Remarks

Note: the order of the component bytes are R, G and B, which is opposite of **COLORREF** type on Windows.

4.9. PaintDataMatrixImageVector

The **PaintDataMatrixImageVector** function writes the DataMatrix barcode in vector image format specified to a disk file.

```
    int __stdcall  PaintDataMatrixImageVector(
        void * pEncodeResult,
        const wchar_t * wszFilename,
        int module_width_hm,
        int target_dpi,
        int forecolor,
        int backcolor,
        int imageType
    );
```

Parameters

pEncodeResult

[in] Pointer that points to the encoder result object.

wszFilename

[in] Pointer to the file name for the image file to be created.

module_width_hm

[in] Module width (X dimension), in the unit of high metric. 1 unit high metric = 1/1000 cm.

target_dpi

[in] The resolution of the target printer.

forecolor

[in] Value of the foreground color, in RGB colorspace.

backcolor

[in] Value of the background color, in RGB colorspace.

imageType

[in] An integer that indicates the image file format. The current version supports three formats: SVG(1), EMF(2) and EPS(3).

Return Values

If the function failed, it returns the error code. You can call function DataMatrixGetErrorMessage to obtain the description of the error.

Remarks

Note: the order of the component bytes are R, G and B, which is opposite of **COLORREF** type on Windows.

4.10. PaintDataMatrixImageRaster2

The **PaintDataMatrixImageRaster** function writes the DataMatrix barcode in raster image format specified to IStream object.

```
int __stdcall  PaintDataMatrixImageRaster2(
    void * pEncodeResult,
    IStream * ostream,
    int pixelsPerModule,
    int forecolor,
    int backcolor,
    int imageType
);
```

Parameters

pEncodeResult

[in] Pointer that points to the encoder result object.

ostream

[in] Pointer to an IStream object.

pixelsPerModule

[in] Number pixels per module with.

forecolor

[in] Value of the foreground color, in RGB colorspace.

backcolor

[in] Value of the background color, in RGB colorspace.

imageType

[in] An integer that indicates the image file format. The current version supports two formats: PNG(0) and BMP(4).

Return Values

If the function failed, it returns the error code. You can call function DataMatrixGetErrorMessage to obtain the description of the error.

Remarks

Note: the order of the component bytes are R, G and B, which is opposite of **COLORREF** type on Windows.

4.11. PaintDataMatrixImageVector2

The **PaintDataMatrixImageVector** function writes the DataMatrix barcode in vector image format specified to an IStream object.

```
int __stdcall  PaintDataMatrixImageVector2(
    void * pEncodeResult,
    IStream * ostream,
    int module_width_hm,
    int target_dpi,
    int forecolor,
    int backcolor,
    int imageType
);
```

Parameters

pEncodeResult

[in] Pointer that points to the encoder result object.

ostream

[in] Pointer to an IStream object.

module_width_hm

[in] Module width (X dimension), in the unit of high metric. 1 unit high metric = 1/1000 cm.

target_dpi

[in] The resolution of the target printer.

forecolor

[in] Value of the foreground color, in RGB colorspace.

backcolor

[in] Value of the background color, in RGB colorspace.

imageType

[in] An integer that indicates the image file format. The current version supports three formats: SVG(1), EMF(2) and EPS(3).

Return Values

If the function failed, it returns the error code. You can call function DataMatrixGetErrorMessage to obtain the description of the error.

Remarks

Note: the order of the component bytes are R, G and B, which is opposite of **COLORREF** type on Windows.

4.12. PaintDataMatrixImageEMF

The **PaintDataMatrixImageClipboard** function Creates a Windows Enhanced Meta File object and returns the handle.

```
HENHMETAFILE __stdcall PaintDataMatrixImageEMF(
    void * pEncodeResult,
        int module_len_hm,
        int target_dpi,
        int forecolor,
        int backcolor,
        int imageType
    );
```

Parameters

pEncodeResult

[in] Pointer that points to the encoder result object.

module_width_hm

[in] Module width (X dimension), in the unit of high metric. 1 unit high metric = 1/1000 cm.

target_dpi

[in] The resolution of the target printer.

forecolor

[in] Value of the foreground color, in RGB colorspace.

backcolor

[in] Value of the background color, in RGB colorspace.

Return Values

If the function failed, it returns the error code. You can call function DataMatrixGetErrorMessage to obtain the description of the error.

Remarks

The caller should destory this handle after using.

4.13. DataMatrixGetErrorMessage

The **DataMatrixGetErrorMessage** function retrieves a string that describes the error.

```
    const char *__stdcall  DataMatrixGetErrorMessage(
        int errorno
    );
```

Parameters

errorno

[in] The error number.

Return Values

a read only string that describes the error.

Remarks

The string returned is a read only string. User should not modify the string directly.

4.14. DataMatrixResultGetBarcodeString2

The **DataMatrixResultGetBarcodeString2** function returns barcode string without requiring user to preallocate the buffer.

```
const char *__stdcall  DataMatrixResultGetBarcodeString2(
    void * pEncodeResult,
    const char * eol
);
```

Parameters

pEncodeResult

[in] The pointer that points to the encoding result object, returned from DataMatrixEncode or DataMatrixEncode2 functions.

eol

[in] Pointer to a NUL terminated string that will be appended to each row in the barcode string.

Return Values

Pointer to a NUL terminated string that represents the barcode (barcode string). If the function failed due to insufficient memory, it returns NULL.

Remarks

The encoder result manages the buffer by itself. Caller should cache the string returned, but not the pointer, as the contents may change after another call of DataMatrixResultBarcodeString2 is made.

4.15. PaintDataMatrixImageClipboard

The **PaintDataMatrixImageClipboard** function Creates a Windows Enhanced Meta File object and place it into the clipboard.

```
int __stdcall  PaintDataMatrixImageClipboard(
    void * pEncodeResult,
    int module_width_hm,
    int target_dpi,
    int forecolor,
    int backcolor
);
```

Parameters

pEncodeResult

[in] Pointer that points to the encoder result object.

module_width_hm

[in] Module width (X dimension), in the unit of high metric. 1 unit high metric = 1/1000 cm.

target_dpi

[in] The resolution of the target printer.

forecolor

[in] Value of the foreground color, in RGB colorspace.

backcolor

[in] Value of the background color, in RGB colorspace.

Return Values

If the function failed, it returns the error code. You can call function DataMatrixGetErrorMessage to obtain the description of the error.

Chapter 5. DataMatrix ActiveX Control Reference

Many programming environments support the use of ActiveX objects. We have tested the Morovia DataMatrix ActiveX Control with Visual Basic, Visual C++, Internet Explorer, Microsoft Word, Excel, Access and IIS programs. The object can be used as a control embedded in a VB form or a dialog, or be used at background for image creation and printing purposes. The DataMatrix ActiveX object can also be inserted into Microsoft Office documents and many other ActiveX-aware programs.

5.1. Specification

Table 5.1. DataMatrix Control Specification

Prog ID	Morovia.DataMatrixControl
ClassID	{16192DED-A048-4136-94C0-2BE735582AD1}
Licensed	no
File name	DataMatrixCtrl.dll
Interface	IDataMatrixCtrlObj
Interface ID	{680D55CA-980E-4559-89A7-F96CA595CE6A}

5.2. Properties

Table 5.2. List of DataMatrix Control Properties

Name	Description
BackColor	Specifies the background color for the control.
ForeColor	Specifies the foreground color for the control.
ModuleWidth	Specifies the width of a module (the smallest unit in a DataMatrix barcode).
Picture	Returns a snapshot of the drawing in Windows Enhanced Metafile Format (EMF).
SizeID	Returns or sets a value for size ID of the data matrix symbols generated.
TargetDPI	Specifies the value of DPI when exporting the images to vector graphics.
Text	Specifies the data to encode.

Table 5.3. List of DataMatrix Control Methods

Name	Description
CopyToClipboard	Place the drawing to clipboard in EMF format.

Name	Description
GetActualSizeID	Retrieves the actual size of the datamatrix symbol generated.
ExportImageVector	Export image to a file in vector graphics format specified (EMF, EPS or SVG).
ExportImageRaster	Export image to a file in raster graphics fromat specified (PNG or BMP).

5.3. BackColor, ForeColor Properties

Description

BackColor - returns or sets the background color of the control.

ForeColor - returns or sets the foreground color of the control.

Syntax

```
object.BackColor[= Color]
object.ForeColor[= Color]
```

Remarks

For open systems we strongly recommend to set the background color to solid white (`0xFFFFFF`) and foreground color to black (`0x000000`). Note: barcode requires decent contrast between the foreground color and the background color in order to be readable. Always test the readability thoroughly when you select a color pair different from black and white.

5.4. ModuleWidth Property

Description

Returns or sets a value that determines the width of a single module in the data matrix symbols generated.

Syntax

```
object.ModuleWidth[= String]
```

Remarks

The “real estate” unit of a DataMatrix symbol, the *module*, is always rectangular. This property sets the width of the rectangle. It affects the overall symbol size.

The default value for **ModuleSize** is `20` mils. The property can be any numbers between `1` and `100`.

This property is a string, and accepts the following units: mil, himetric, mm, cm, pt and inch. For example, “0.01cm” and “1mm” are all valid. If no measure unit is specified, unit of mil is assumed.

5.5. Picture Property

Description

Readonly property. Returns a snapshot of the drawing in Windows Enhanced Metafile Format (EMF).

Syntax

```
object.Picture
```

Remarks

The *Picture* property provides a convenient method to retrieve the drawing without first saving it to disk. The picture object contains an enhanced metafile handle which can be passed to *clipboard* or played on a device.

The included VB6, VC++ and C# examples use this property to retrieve an EMF handle and play it on the target printer to print the barcode.

5.6. SizeID Property

Description

Gets and Sets the size ID desired.

Syntax

```
object.SizeID
```

Remarks

Use this property to pass the datamatrix size desired, and the modifier byte that specifies additional encoder parameters. See Appendix C, *Size ID Parameter (updated in version 5.1)* for more information.

5.7. TargetDPI Property

Specifies the resolution of the target printer, applicable during exporting vector graphics.

Syntax

```
object.TargetDPI = Number
```

Remarks

Whe producing small sizes of barcodes, especailly on lowe resolution printers, lengths must be align to the boundary of pixels in order to retain high quality. The ActiveX control adjusts the drawing units automaticaly based on this property when exporting vector graphics images.

5.8. Text Property

Specifies the Data to be encoded into the DataMatrix symbol.

Description

Returns or sets a string for the message to be encoded.

Syntax

```
object.Text[= String]
```

Remarks

5.9. CopyToClipboard Method

Copies the DataMatrix image into the system clipboard, in EMF format.

Syntax

```
obj.CopyToClipboard
```

Description

Use this method to place a copy of barcode image to the clipboard, so that you can either retrieve it immediately in your application, or other applications such as Microsoft Word.

5.10. ExportImageRaster Method

Description

Exports the current drawing to a disk file, in raster image format (PNG or BMP).

Syntax

```
obj.ExportImageRaster(filename, pixelsPerModule, imageType)
```

Parameters

filename

[in] filename. A string that corresponds to the file to be written.

pixelsPerModule

[in] Number of pixels per module.

imageType

[in] Image type. The current version supports two raster image formats:

- 0 - PNG
- 4 - BMP

5.11. ExportImageVector Method

Description

Exports the current drawing to a disk file, in raster image format (EMF, SVG and EPS).

Syntax

```
obj.ExportImageVector(filename, imageType)
```

Parameters

filename

[in] filename. A string that corresponds to the file to be written.

imageType

[in] Image type. The current version supports two three image formats:

- 1 - SVG
- 2 - EMF
- 3 - EPS

5.12. GetActualSizeID Method

Description

Retrieves the actual value of SizeID in the DataMatrix barcodes generated.

Syntax

```
obj.GetActualSizeID()
```

Remarks

The property SizeID returns the size ID that you set. Use this method to retrieve the size ID in the datamatrix symbol created.

Chapter 6. Adding DataMatrix to Crystal Reports

Adding DataMatrix barcodes to Crystal Reports is quite simple. The software includes a report file authored in Crystal Reports 9.

Note: the functions in this release are not backward compatible with old version DataMatrix Fontware 3. You need to change the functions in order to use the new version API.

> **Warning** Due to the length constraint imposed by Crystal Reports, the approach outlined in this chapter requires Crystal Reports version 9 and above. If you need to work with a lower version, write to `support@morovia.com` to request a trial of version 3.

6.1. UFL Functions

A Crystal Reports extension DLL is included in the software (`cruflDataMatrix5.dll`), which provides data matrix encoding functions. By default, this file can be found under the installation directory. This is a COM DLL - so you must register it if you move the file to another computer. This is typically done by running `regsvr32 cruflDataMatrix5.dll` command in a administrator console. In this release, both 32-bit and 64-bit versions are supplied. The 32-bit version is required to work with Crystal designer. The 64-bit is needed to work with 64-bit Crystal runtime. On 64-bit Windows, both DLLs are installed. [1]

This DLL exports two functions:

Table 6.1. List of Crystal Reports UFL Functions (`cruflDataMatrix5.dll`)

Function Name	Comment
Number DataMatrixEncodeSet (String, Number)	Returns the number of chunks required for the data encoded. The second parameter specifies the size ID.
Number DataMatrixEncodeSetRAS (String, Number)	If you are using RAS print engine, use this encoder function instead of DataMatrixEncodeSet to work around a bug in RAS. See KB10052[2] for details.
String DataMatrixEncodeGet(Number)	Returns a chunk. Each chunk consists of no more than 254 characters (the maximum number allowed by Crystal Reports).

Function `DataMatrixEncodeGet` takes one argument which is the ordinal number of the chunk. Function `DataMatrixEncodeSet` has two parameters. The first parameter is data encoded, followed by the size ID. For the meaning of size ID parameter, refer to Section 2.1, “What is Data Matrix?”. This function returns a number, which indicates the number of subsequent calls to `DataMatrixEncodeGet`. Each call of `DataMatrixEncodeGet` retrieves a chunk of encoder results.

The following script demonstrates the basic flow to get it work in Crystal Reports:

[1] The 64-bit DLL is installed under `C:\Program Files\Morovia DataMatrix Fonts & Encoder 5` directory, while the 32-bit DLL can be found under `C:\Program Files (x86)\Morovia DataMatrix Fonts & Encoder 5`.

```
//Mroovia example formula to display Datamatrix barcodes
StringVar CompleteBarcodeString:="";
StringVar DataToEncode:= {Your Field Here};
NumberVar i:=0;
NumberVar Segments:=
DataMatrixEncodeSet(DataToEncode, -1);
For i:=0 to Segments Do
(
   CompleteBarcodeString := CompleteBarcodeString +
             DataMatrixEncodeGet(i);
);
CompleteBarcodeString
```

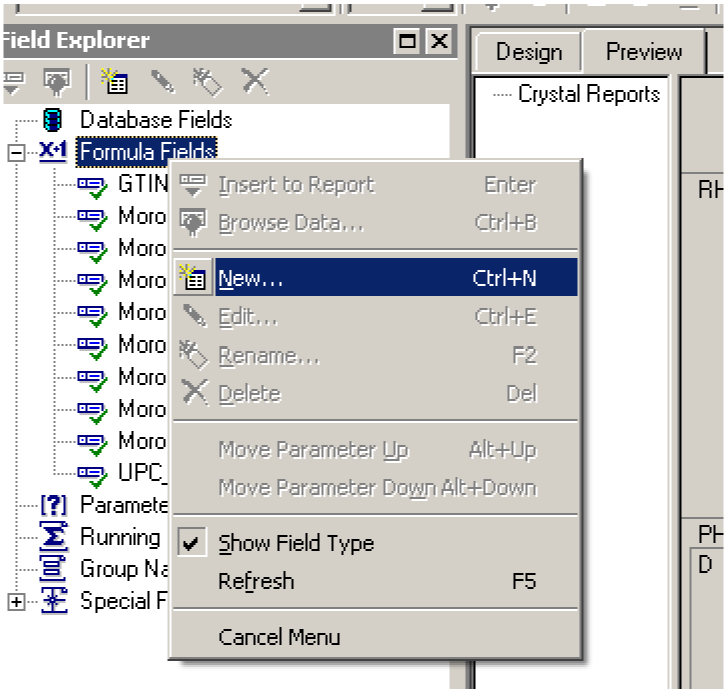
Because of the length restraint that UFL imposes on the string, it is not possible to complete in one UFL call. Instead, you prepare all parameters and call `DataMatrixEncodeSet`. This function calls encoder at the back end, and returns the number of chunks required. After that the program enters a loop that calls `DataMatrixEncodeGet` repeatedly until all result are retrieved and stored in `BarcodeString`.

6.2. Tutorial: Step by Step

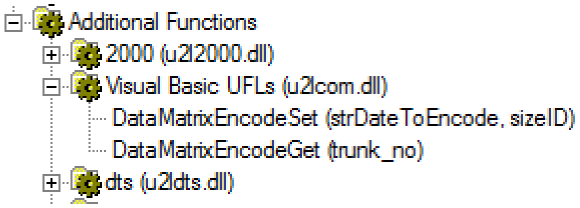
The screen shots in the following tutorial were taken in Crystal Reports 9. Interfaces in other versions may appear slightly differently.

Assume that you have a database table with several fields. You already put them into the report, and want to add a column to hold DataMatrix barcode for ID field.

1. First switch to design mode. Choose Report → Formula Workshop.
2. Right click on Formula Fields and choose New.
3. Give your formula field a name, in this example we will name it `Morovia_DataMatrix`. In versions 9 and above, you will be prompted to use the editor or the expert, choose Use Editor.

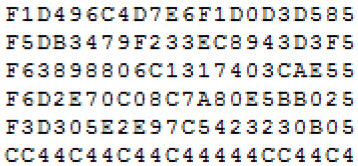


4. In the Formula Editor, choose Functions → Additional Functions → Visual Basic UFLs (u2lcom.dll). You should see `DataMatrixEncodeSet` appears under. If not, the DLL is not installed properly.



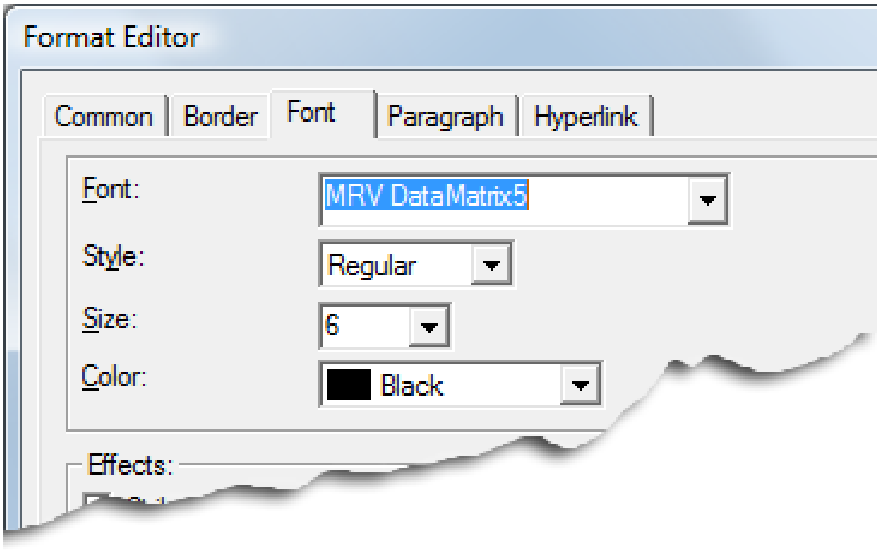
```
Additional Functions
    2000 (u212000.dll)
    Visual Basic UFLs (u2lcom.dll)
        DataMatrixEncodeSet (strDateToEncode, sizeID)
        DataMatrixEncodeGet (trunk_no)
    dts (u2ldts.dll)
```

5. In the edit box below, copy and paste entire script in Table 6.1, “List of Crystal Reports UFL Functions (`cruflDataMatrix5.dll`)”. Make necessary changes. For example, you are likely to change the value of `DataToEncode` variable to one of your database field. Version, Error Correction Level can be changed in the code accordingly.
6. Choose Save → Close to close Formula Workshop.
7. From the Field Explorer, drag the `Morovia_DataMatrix` formula field to the report.
8. Choose File → Print Preview. You should see a series of meaningless lower case letters and digits in the box. This is normal as the barcode string is completely difference from the data encoded in case of DatatMatrix.



```
F1D496C4D7E6F1D0D3D585
F5DB3479F233EC8943D3F5
F63898806C1317403CAE55
F6D2E70C08C7A80E5BB025
F3D305E2E97C5423230B05
CC44C44C44C44444CC44C4
```

9. Switch back to design mode. Right click on the field and choose Format Field. Click on the Font tab, and choose the `MRV DataMatrix5` font. Set the font size to 6 points or to the size appropriate for your environment.



Font: MRV DataMatrix5

Style: Regular

Size: 6

Color: Black

10. Drag the cursor to make the formula field big enough to contain the entire barcode. You may need to adjust both horizontally and vertically. Be sure to leave some spaces for quiet zone requirement.

11. Run the Print Preview again. The barcode should come up. Print a page and scan the barcode to make sure that the font size is appropriate and the bounding rectangle is big enough to hold the whole barcode.

Note If you add barcodes to report using the trial version and subsequently upgrade to the full version, you need to refresh the report once. Otherwise the barcode will continue to be scrambled.

6.3. Distributing UFL, Fonts with your report application

Once you finish the report design, you can distribute your report application with Crystal run time files, barcode fonts and the UFL library.

Note Developer License is required to distribute font files and encoder UFL DLL outside your organization.

Two run time files need to be included in your distribution package:

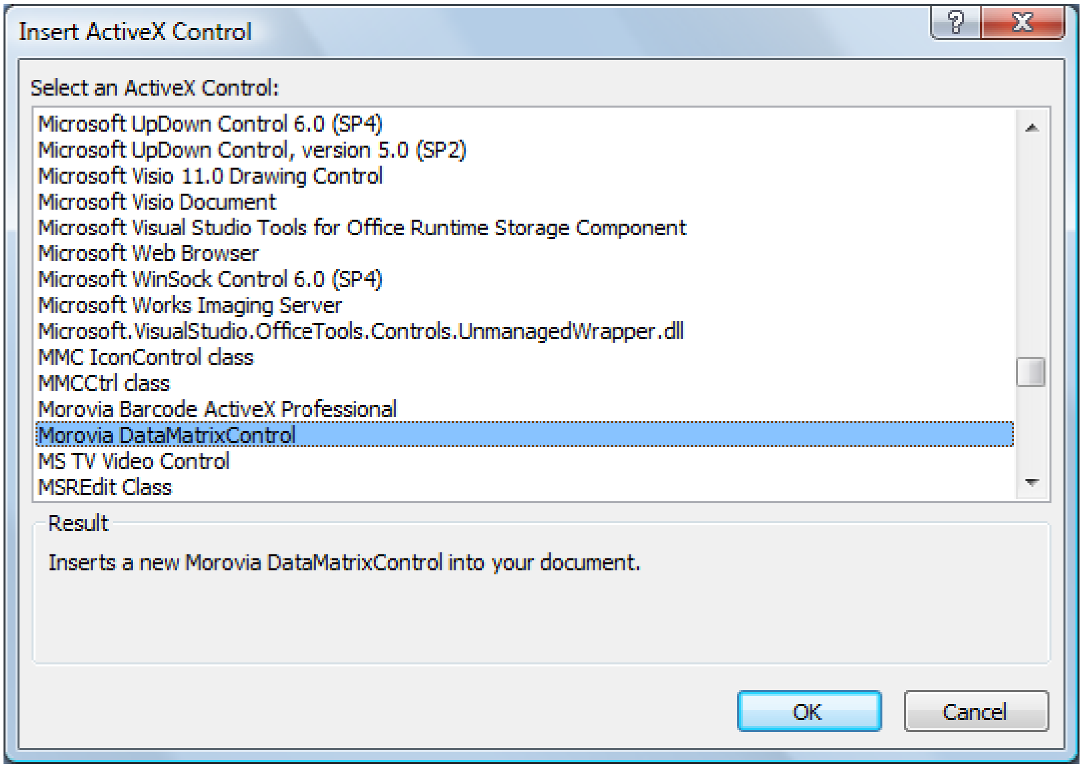
1. **Morovia DataMatrix Font**. The font file is located `C:\windows\fonts\mrvdatamatrix5.ttf`, and should be copied to the target computer's `fonts` folder.
2. **DataMatrix Encoder UFL**. This DLL is installed at `C:\Program Files\Morovia DataMatrix Fonts & Encoder 5` by default. In the production computer, you can place this DLL anywhere. However, this is a COM DLL and requires registration. If you are deploying to a 64-bit Crystal Reports runtime, you need the 64-bit version of the DLL.

Chapter 7. Adding DataMatrix to Microsoft Office Documents

7.1. Access (ActiveX Control)

This section explains the steps to add DataMatrix barcodes in a Microsoft Access report using Morovia DataMatrix ActiveX Control.

1. The DataMatrix ActiveX control must be installed and registered on the computer.
2. Open a report in design view and choose Insert → ActiveX Control.... If you are using Access 2007 or 2010, switch to Design view, and click on Insert ActiveX Control button.
3. From the list of controls, select `Morovia DataMatrixControl`.



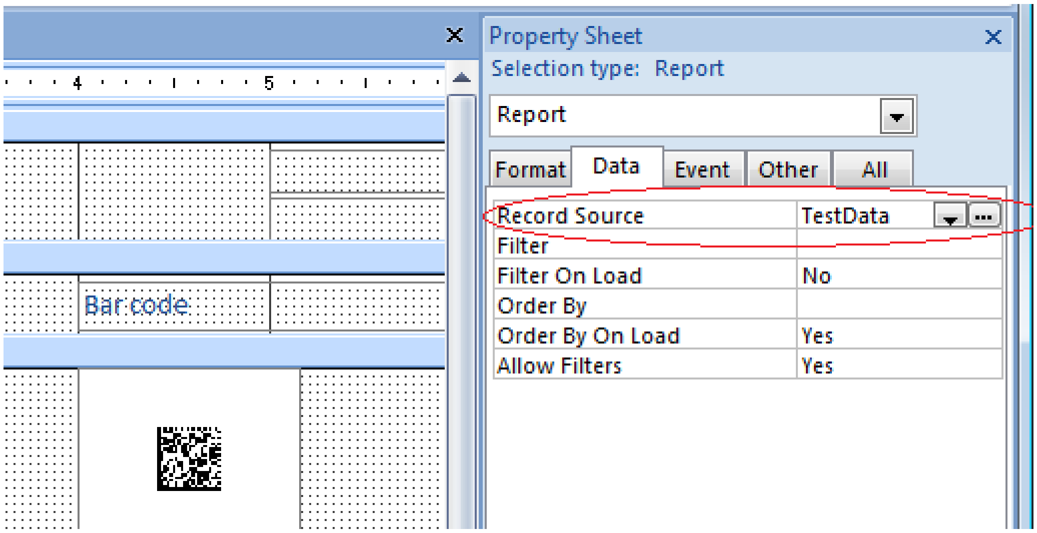
Select an ActiveX Control:

Microsoft UpDown Control 6.0 (SP4)
Microsoft UpDown Control, version 5.0 (SP2)
Microsoft Visio 11.0 Drawing Control
Microsoft Visio Document
Microsoft Visual Studio Tools for Office Runtime Storage Component
Microsoft Web Browser
Microsoft WinSock Control 6.0 (SP4)
Microsoft Works Imaging Server
Microsoft.VisualStudio.OfficeTools.Controls.UnmanagedWrapper.dll
MMC IconControl class
MMCCtrl class
Morovia Barcode ActiveX Professional
Morovia DataMatrixControl
MS TV Video Control
MSREdit Class

Result

Inserts a new Morovia DataMatrixControl into your document.

4. After the control is placed on the report, right click on it and choose Properties...
5. Modify the Control Source property to point to the table and field of the data you wish to encode into the barcode. The data source can be a table field, or a field of a comprehensive query result.



Property Sheet

Selection type: Report

Report

Record Source	TestData
Filter	
Filter On Load	No
Order By	
Order By On Load	Yes
Allow Filters	Yes

6. Modify other properties, such as `ModuleWidth`, and `SizeID`. After you are satisfied with the result, close the property dialog.
7. Save the run the report. You should see the barcodes appear in the report.

7.2. Access (using DataMatrix Font)

You can also add DataMatrix barcodes to Access report using font-based solution.

1. Before creating barcodes in Microsoft Access, you must import the required module. This module adds VBA function so that you can put into the report. Choose Modules → Import and select `Morovia.DataMatrixFontDLL5.bas` file, located in the program folder.
2. Open a report in design view and add a text box to the report. The text box will be modified to contain a barcode.
3. Right click on the text box and choose properties.
4. Place the formula `=DataMatrixEncode([TestData.Data],-1)` in the control source property of the text box where [TestData.data] is the field that contains the data to be encoded into the DataMatrix barcode. The following parameter is the size ID desired.
5. Run the report. You should see lines of hexadecimal characters appear in the place of the text box. This is the barcode string in the raw text form.
6. Go back to the design view and change the font of the text box. In our case, choose `MRV DataMatrix5` and 6 points. Adjust the size of the text box to fit the whole barcode.
7. Save and run your report. You should see the barcodes appear in the report.

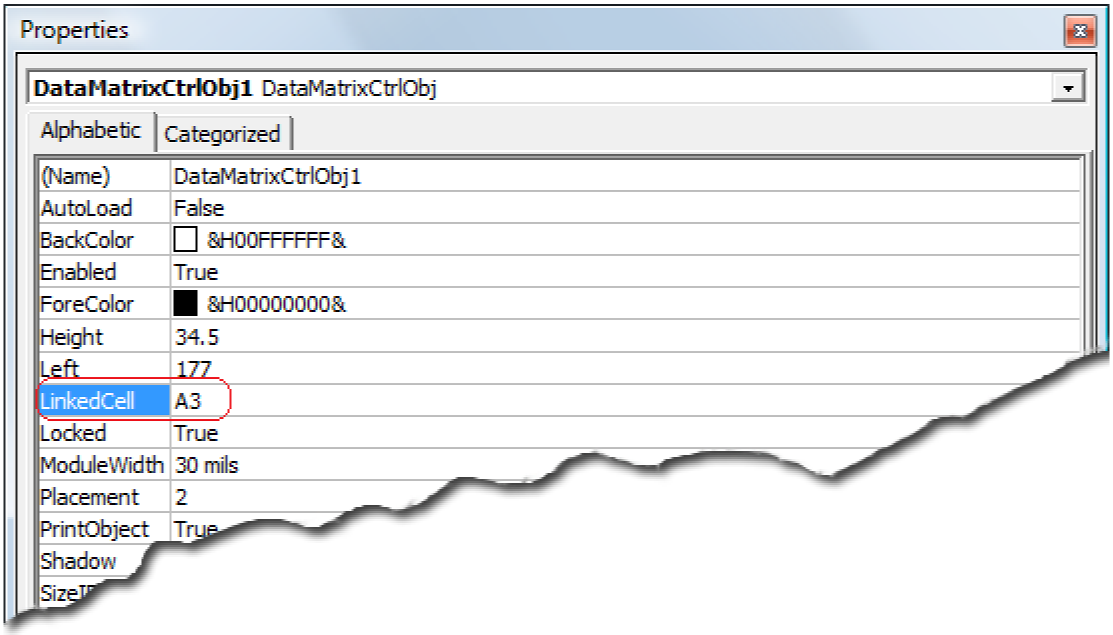
7.3. Excel (ActiveX Control)

Excel has line gap issues with DataMatrix fonts. You can paste EMF image from GUI encoder, or use ActiveX control as outlined below.

1. After you finished other parts of the spreadsheet, choose View → Toolbars → Control Toolbox
2. When Control Toolbox appears, click on More Controls button.

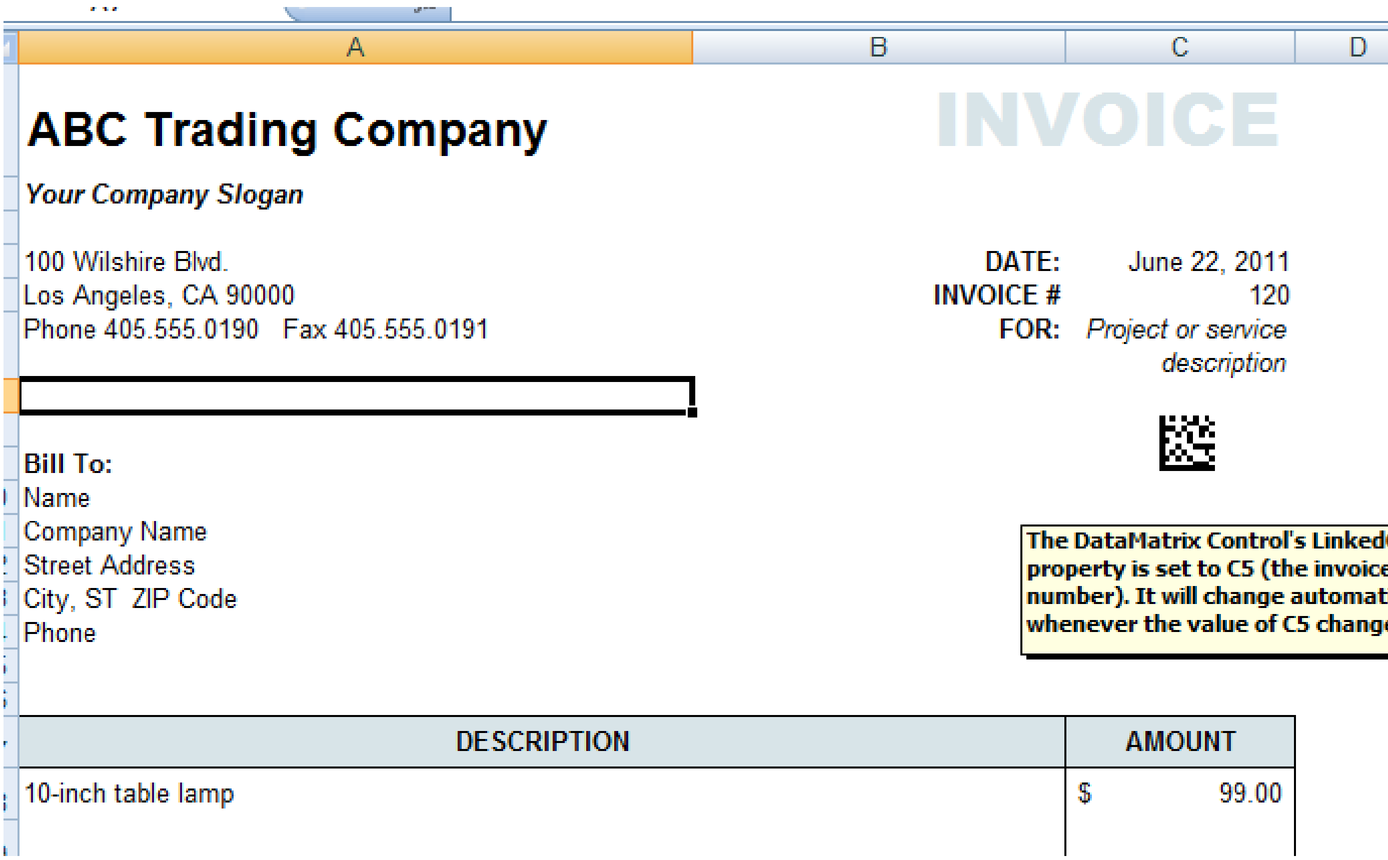
On Excel 2007 and 2010, first switch to developer tab, then press Insert button on the toolbar, and select More Controls button.

3. From the list of controls presented, choose Morovia DataMatrixControl.
4. Select the area to place the control in the spreadsheet.
5. Right click on the control, choose Properties and change the Linked Cell property to the name of the cell that contains the data you wish to encode.



(Name)	DataMatrixCtrlObj1
AutoLoad	False
BackColor	□ &H00FFFFFF&
Enabled	True
ForeColor	■ &H00000000&
Height	34.5
Left	177
LinkedCell	A3
Locked	True
ModuleWidth	30 mils
Placement	2
PrintObject	True
Shadow	
Size	

6. Change other properties as necessary such as `Rows`, `Cols` and `AspectRatio` to adjust the size of the barcode.
7. After editing the properties, click on Exit Design Mode button to exit design mode. The barcode will appear in the spreadsheet.
8. The barcode is now bound to the cell. Change the data of the linked cell, the barcode will change accordingly.



ABC Trading Company

INVOICE

Your Company Slogan

100 Wilshire Blvd.
Los Angeles, CA 90000
Phone 405.555.0190 Fax 405.555.0191

DATE: June 22, 2011
INVOICE # 120
FOR: Project or service description

Bill To:
Name
Company Name
Street Address
City, ST ZIP Code
Phone

The DataMatrix Control's Linked
property is set to C5 (the invoice
number). It will change automat
whenever the value of C5 chang

DESCRIPTION	AMOUNT
10-inch table lamp	$ 99.00

Note: to subsequently modify or delete the barcode control, Excel must enter `Design Mode`. This can be done by pressing the Design Mode button on the Control Toolbox.

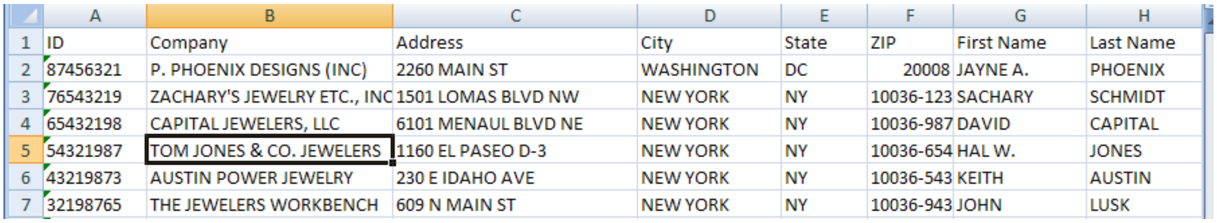
7.4. Microsoft Word

Using Datamatrix control in Microsoft Word is similar to the one in Excel, except that Word does not provide a way for data binding.

1. choose View → Toolbars → Control Toolbox.
2. In the toolbox, choose the more controls button.
3. Select Morovia DataMatrixControl from the list of available ActiveX controls. After selecting it, the control will appear in the document, the control may be sized as necessary. To change the properties of the control right click on the control and choose Properties.
4. When finished, exit the design mode by choosing the design mode button.
5. To edit the properties of the control the program must be in design mode. If there are problems editing the properties of the control, press the design mode button to enable it.

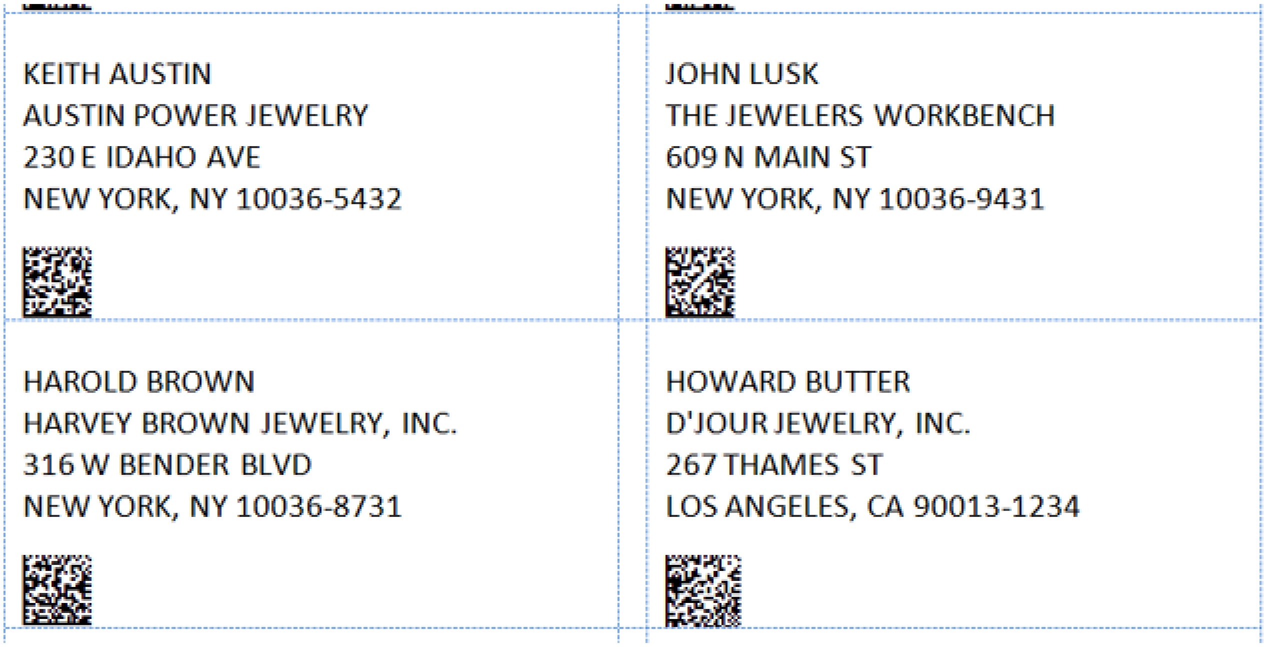
7.5. Word Mail Merge

This tutorial uses Excel file `Word Mail Merge DataSource.xls` as mail merge data source. The data looks like this:



	A	B	C	D	E	F	G	H
1	ID	Company	Address	City	State	ZIP	First Name	Last Name
2	87456321	P. PHOENIX DESIGNS (INC)	2260 MAIN ST	WASHINGTON	DC	20008	JAYNE A.	PHOENIX
3	76543219	ZACHARY'S JEWELRY ETC., INC	1501 LOMAS BLVD NW	NEW YORK	NY	10036-123	SACHARY	SCHMIDT
4	65432198	CAPITAL JEWELERS, LLC	6101 MENAUL BLVD NE	NEW YORK	NY	10036-987	DAVID	CAPITAL
5	54321987	TOM JONES & CO. JEWELERS	1160 EL PASEO D-3	NEW YORK	NY	10036-654	HAL W.	JONES
6	43219873	AUSTIN POWER JEWELRY	230 E IDAHO AVE	NEW YORK	NY	10036-543	KEITH	AUSTIN
7	32198765	THE JEWELERS WORKBENCH	609 N MAIN ST	NEW YORK	NY	10036-943	JOHN	LUSK

We want to print address information as well as a DataMatrix barcode that encodes such information in Avery label paper 5163. The Print Preview looks like the one below:



KEITH AUSTIN
AUSTIN POWER JEWELRY
230 E IDAHO AVE
NEW YORK, NY 10036-5432

JOHN LUSK
THE JEWELERS WORKBENCH
609 N MAIN ST
NEW YORK, NY 10036-9431

HAROLD BROWN
HARVEY BROWN JEWELRY, INC.
316 W BENDER BLVD
NEW YORK, NY 10036-8731

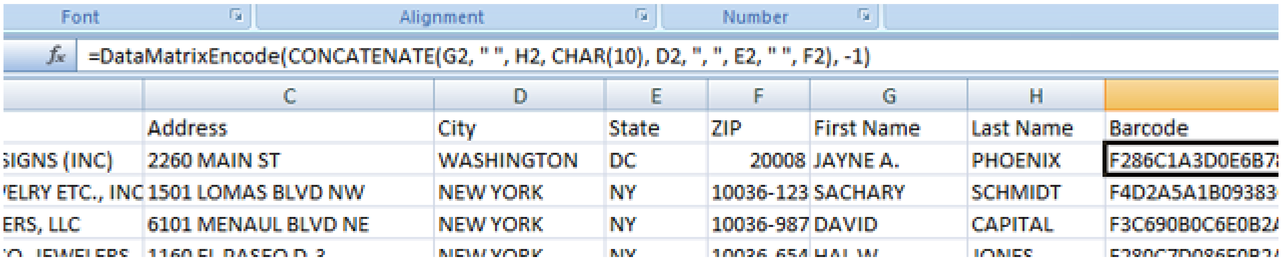
HOWARD BUTTER
D'JOUR JEWELRY, INC.
267 THAMES ST
LOS ANGELES, CA 90013-1234

1. Before we start, we need to import a module into Excel. To do that, open Visual Basic Editor. In Excel 2007, this is done by selecting Developer → Visual Basic.

   In Visual Basic Editor, choose File → Import File. Navigate to the DataMatrix Fonts & Encoder 5 installation folder, and select the `Morovia.DataMatrixFontDLL5.bas`.

   Close Visual Basic Editor.

2. Add a new column in the spreadsheet that will hold the barcode string. In our case, we use column H. In cell H2, enter the definition as below:



```
=DataMatrixEncode(CONCATENATE(G2, " ", H2, CHAR(10), D2, ", ", E2, " ", F2), -1)
```

	C	D	E	F	G	H	
	Address	City	State	ZIP	First Name	Last Name	Barcode
SIGNS (INC)	2260 MAIN ST	WASHINGTON	DC	20008	JAYNE A.	PHOENIX	F286C1A3D0E6B7
'ELRY ETC., INC	1501 LOMAS BLVD NW	NEW YORK	NY	10036-123	SACHARY	SCHMIDT	F4D2A5A1B09383
ERS, LLC	6101 MENAUL BLVD NE	NEW YORK	NY	10036-987	DAVID	CAPITAL	F3C690B0C6E0B2
O. JEWELERS	1160 EL PASEO D-3	NEW YORK	NY	10036-654	HAL W.	JONES	F280C7D086E0B2

After hitting **Enter**, you should see a hexadecimal string result show up. If not, examine the formula you entered.

Note the use of Excel function `CONCATENATE` here. This function is used to combine several fields, as well as line return characters.

See the DLL API for the meaning of each field. Here we set size ID to -1.

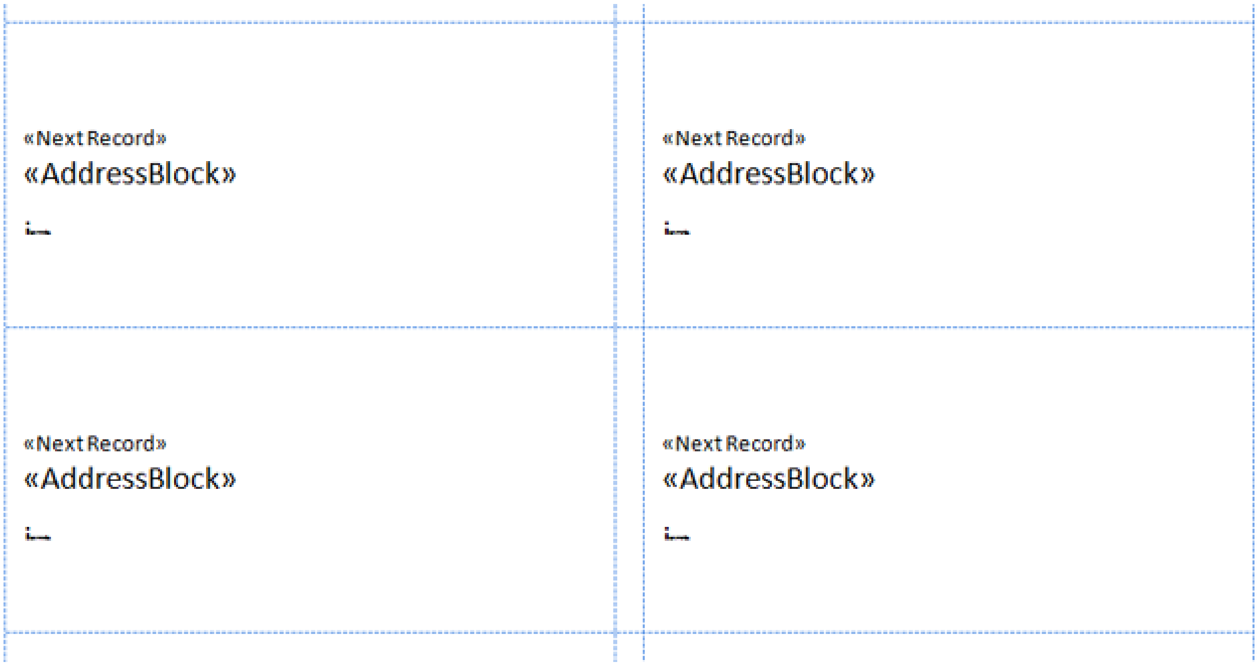
3. Copy the formula to other cells of the same column. This can be done by selecting cell H2, highlighting the cells that the formula is copied, and select Paste.

   Close the Excel file and start Microsoft Word.

4. In Microsoft Word, choose Mailing → Start Mail Merge → Labels. Select Avery 5163 as the label we will work on.
5. Select Select Recipients Select Existing List. In the file dialog, navigate to the spreadsheet we just created. If it asks for Select Table, choose Sheet1$.
6. Click on Address Block. This is for the address line. Microsoft Word has the intelligence to select the address block.
7. Hit **Enter** once to move the cursor below, and click on Insert Merge Field. Select Barcode as the merge field.

   Click on Review Results. You should see the first record show up, with hexadecimal characters in the place of barcode.

8. Now adjust the font for the address block and the barcode field. For the address block, we use Arial 16 points. For the barcode, use `MRV DataMatrix5` 6 points.
9. Copy the format to other labels by selecting Update labels.


«Next Record»
«AddressBlock»

«Next Record»
«AddressBlock»

«Next Record»
«AddressBlock»

«Next Record»
«AddressBlock»

10.Select Preview Results to view the sheets of labels.

Chapter 8. Adding DataMatrix Symbols to SQL Server Reporting Service

Adding data matrix barcodes to SQL Reporting Service report becomes an easy task after you installed DataMatrix Font & Encoder 5. This chapter explains how you can achieve the objective. Our report service plugin supports SQL Reporting Service 2000, 2005 and 2008. This chapter uses Visual Studio 2008/SQL Reporting Service 2008 for demonstration. To follow the steps, you also need to have AdventureWorks sample database installed.

> **Note** This chapter contains instructions of working on SSRS 2008 and prior versions. For working with SQL Server Reporting Service 2010 or above, visit KB10643[1].

8.1. Custom Assembly

SSRS can't use the encoder DLL directly. A ready-to-use custom assemblies built under Visual Studio 2005 (.net 2.0) is included in the software.

> **Note** In case that you need to build the assembly by yourself, The source code is also included in the distribution. The project files are packed in `ReportServicePlugin.zip`, which is located under the program (x86) folder.

This assembly exposes single function: `DataMatrixEncode`. The prototype for this function is as below:

```
string DataMatrixEncode(string strDataToEncode,
    int sizeIDRequested);
```

The function is quite simple: the first parameter is the data encoded, followed by the size ID requested. The function returns the barcode string if succesful, otherwise `ERROR` is returned.

8.1.1. Installing Custom Assembly

For Reporting Service 2000, copy the DLL into the following two directories:

- Report Designer Folder: `[Program Files]Microsoft SQL Server\80\Tools\Report Designer`.
- Deploy folder `[Program Files]Microsoft SQL Server\MSSQL\Reporting Services \ReportServer\bin`.

For Reporting Service 2005 or 2008, copy the DLL into the following directories:

- Report Designer Folder: `[Program Files]Microsoft Visual Studio 9.0\Common7\IDE \PrivateAssemblies`.
- Deploy folder `[Program Files]Microsoft SQL Server\MSSQL\Reporting Services \ReportServer\bin`.

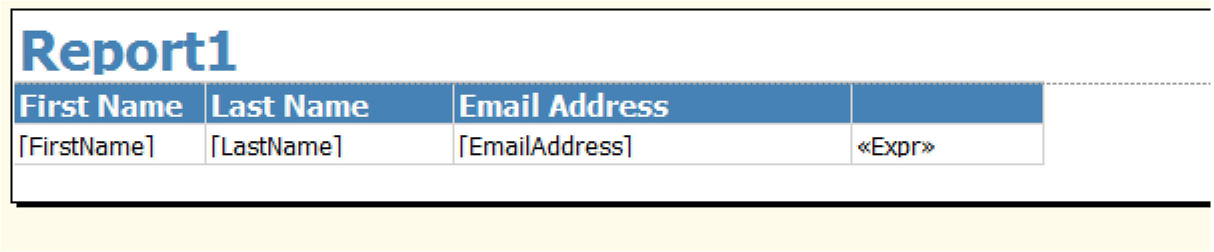
[1] http://www.morovia.com/kb/10643

If you have multiple SQL Server instances, you will have multiple `MSSQL.n` directories. Locate the one that the Reporting Service is installed under.

8.2. Adding DataMatrix Barcodes to Report

In this tutorial, we demonstrate how you can add data matrix bacodes into a report. The data table is `Person.Contact`, available in `AdventureWorks` sample database. We'd like to present email address in datamatrix barcode.

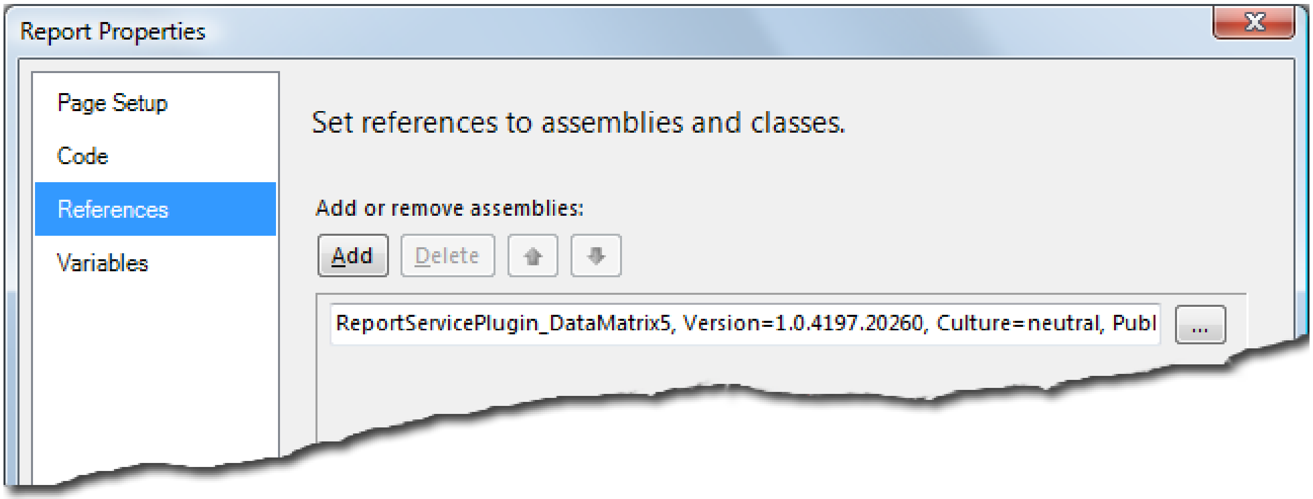
1. Open Visual Studio 2008 or SQL Server Business Intelligence Development Studio and create a new Report Server Project.
2. In Query builder, enter SQL statement `SELECT * FROMPerson.Contact`.
3. Layout the report as desired. Make changes to the format of each field. Add a new column to the right that will hold the barcodes. The result looks like below:



Report1

First Name	Last Name	Email Address	
[FirstName]	[LastName]	[EmailAddress]	«Expr»

4. Select Report → Report Properties to bring up Report Properties dialog.
5. click on References tab. Navigate to the location of the customer assembly and add it.



Report Properties

Set references to assemblies and classes.

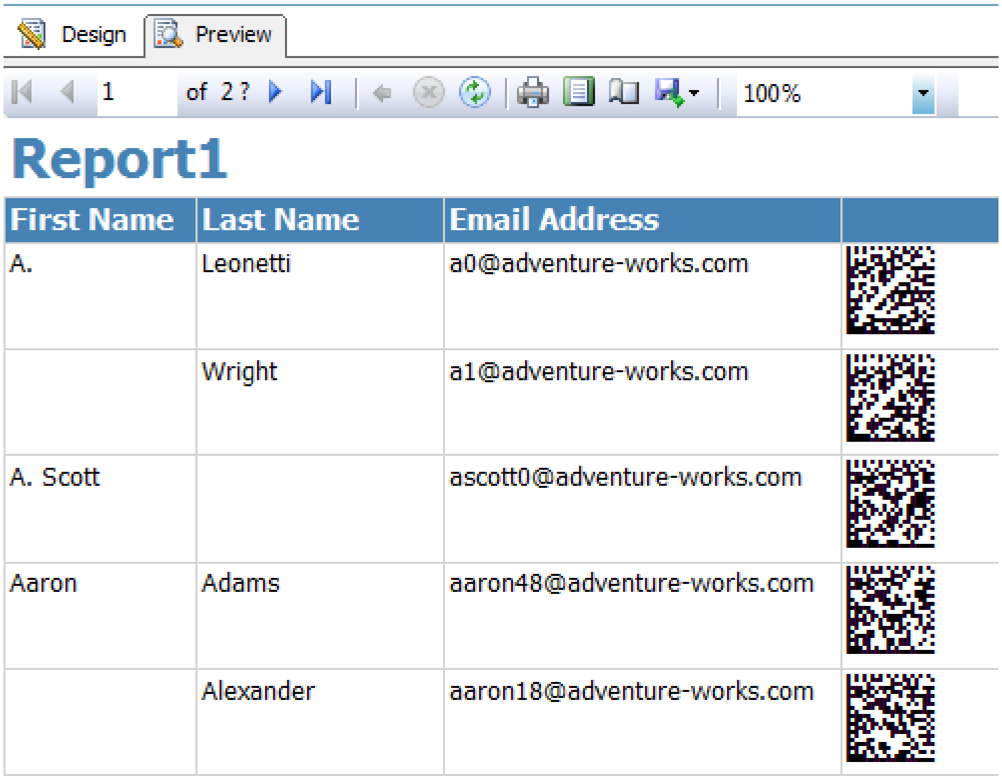
Add or remove assemblies:

ReportServicePlugin_DataMatrix5, Version=1.0.4197.20260, Culture=neutral, Publ

6. Click on the new column created, and select expressions, change its value to the formula below:

   ```
   ==Morovia.ReportService.DataMatrixV5.DataMatrixEncode(Fields!EmailAddress.Value,-1)
   ```

   The formula calls the DataMatrixEncode function with automatic size selection.
7. Now preview the report. You should see lines of hexadecimal characters at the place of the barcode.
8. Format the text box with `MRV DataMatrix5`, 6 points. Preview the report again, you should see the barcodes.



Report1

First Name	Last Name	Email Address	
A.	Leonetti	a0@adventure-works.com	
	Wright	a1@adventure-works.com	
A. Scott		ascott0@adventure-works.com	
Aaron	Adams	aaron48@adventure-works.com	
	Alexander	aaron18@adventure-works.com	

If you are printing to a laser printer, you should get a good quality barcode label printed. Now if you are printing to a low resolution thermal printer, such as a Zebra with 203 dpi in resolution, you should examine if the font size produces the optimal results. Follow the steps below:

1. According to Table 2.2, “DataMatrix Font Characteristics”, `MRV DataMatrix5` produces barcodes with X dimension at 20 mils at 6 points. Now translate it into inches and multiply the result by the printer resolution: 20x0.001*203=4.
2. 4 pixels is optimized value. Therefore we can use 6 points on 203 thermal printer.

8.3. Deployment

The Reporting Service require any custom assembly defined in the security policy otherwise a run-time error will be thrown and all you get is #Error without any explanation. Follow the steps below to change security policy. Two security policy files are required to change:

- `RSPreviewPolicy.config`. This policy file is used for `DebugLocal` preview in Visual Studio. This file is located in the Report Designer folder which is `[Program Files]Microsoft SQL Server\80\Tools\Report Designer` for RS2000 and [Program Files]Microsoft Visual Studio 9.0\Common7\IDE\PrivateAssemblies for RS2005.
- `rssrvpolicy.config`, the policy file used for running Report Server. The file is located under `[Program Files]Microsoft SQL Server\MSSQL\Reporting Services\ReportServer\bin` directory.

Perform the following steps to add security policy required to run custom assembly:

1. Open `RSPreviewPolicy.config`, and add the following content at the end (just before two ending `CodeGroup` tags):[2]

```
<CodeGroup class="FirstMatchCodeGroup"
```

```
  version="1"
  PermissionSetName="FullTrust"
  Name="ReportServicePlugin_DataMatrix5.dll" Description="ReportServicePlugin_DataMatrix5.dll">
  <IMembershipCondition class="UrlMembershipCondition" version="1"
Url="C:\Program Files\Microsoft Visual Studio 9.0\Common7\IDE\" \
"PrivateAssemblies\ReportServicePlugin_DataMatrix5.dll" />
</CodeGroup>
```

Note that on RS2000 the custom assembly is located in a different directory.

2. Save the file. In Visual Studio, change the active configuration to `DebugLocal` and run the report. You should see the barcodes on the report. Examine the contents in the Output window.

   If you see message A first chance exception of type 'System.Security.SecurityException' occurred in mscorlib.dll, the security is not configured properly.

3. After you have successfully run the report under `DebugLocal` configuration, publish the report to the Report Server. Open `rssrvpolicy.config` and add similar lines.

```
<CodeGroup class="FirstMatchCodeGroup"
  version="1"
  PermissionSetName="FullTrust"
  Name="ReportServicePlugin_DataMatrix5.dll" Description="ReportServicePlugin_DataMatrix5.dll">
  <IMembershipCondition class="UrlMembershipCondition" version="1"
Url="C:\Program Files\Microsoft SQL Server\MSSQL.2\Reporting Services\" \
"ReportServer\bin\ReportServicePlugin_DataMatrix5.dll" />
  </CodeGroup>
```

   Note that you may change the file path if it is located in a different location.

4. Restart SQL Server Reporting Service and browse the report. You should see the barcodes on the report this time.

Appendix A. Technical Support

Morovia offers a wide variety of support services. To help you save time and money when you encounter a problem, we suggest you try to resolve the problem by following the options below in the order shown.

- Consult the documentation. The quickest answer to many questions can be found in the Morovia product documentation.
- Review the tutorial and sample applications. The tutorial steps you through the development process for a typical barcode application. The sample applications provide working code examples in several programming languages. All sample applications are extensively commented.
- Access Morovia Online. Morovia Online provides a knowledge base which documents the frequently asked questions and a web forum.

  The web address for knowledge base is http://support.morovia.com. You can ask question at support forum at http://forum.morovia.com.
- Contact Morovia Technical Support Service. The Technical Support service is provided for free up to 180 days after the purchase. Email Morovia support engineers at `support@morovia.com`.

Note If you purchased your software from our reseller, check to see if they provide support services before contacting Morovia.

Support services and policies are subject to change without notice.

Appendix B. Input format (ECI and Structural Append)

Data matrix encoder accepts three parameters - the data encoded, the target size, and the line feed string. The target size is an integer representing one of 30 choices that a data matrix symbol can have, plus value 0 for automatic size selection. The line feed string is used to separate lines so you can parse the result into an array of strings, with each representing a barcode row.

The data encoded is a single-byte ASCIIZ string. The control character NULL (the first one in the ASCII table) is not allowed in an ASCIIZ string, because this peculiar character is reserved as the indicator of end of string (EOS) in C language. If you want to encode this character, you need to use the escaped format.

Escaping means using an alternative form (usually a combination of several readable characters) to enter the character instead. It is necessary since some characters are not normally acceptable due to the limitations imposed by the programs or transport protocols, such as control characters (ASCII value 0 - 31).

There are several advanced data matrix features which require special input. Because all character values, from 0 to 256 can be encoded into a data matrix symbol, the only way to enter them into the input parameter is through the escaped form. These features include Macro 5 and 6, ECI and Structural Append.

> **Note** If you are not encoding control characters, or not using any of the advanced features, you can skip this chapter all together.

B.1. Tilde codes

Tilde codes always start with the ASCII character tilde ~ (character value 126). The corresponding key on a standard PC keyboard sits at the top left corner, just under **ESC** key.

Although many vendors support it, the tilde codes are not public standards. We have not seen any endorsement by standards organizations. It does not appear in the data matrix standard either.

~dnnn

When nnn corresponds to a numeric value between 0 and 255, the tilde code sequence represents a character with value equal to nnn. For example, ~d032 represents a space character.

~~

Represents a tilde (~) character.

~1

Represents a FNC1 character. The tilde escape sequence can appear anywhere in the input.

~2

Indicates that a structural append control block follows. See structural append topic for details.

~3

Represents a symbol character which means that message followed is used for reader programming. This escape sequence must appear at the beginning of the input.

~5

Represents a symbol character which encodes Macro 5 abbreviation. Must appear at the beginning of the message.

~6

Represents a symbol character which encodes Macro 6 abbreviation. Must appear at the beginning of the input.

~7

Indicates the start of an ECI block. This escape sequence must be followed by exact 6 digits, which corresponds to the ECI value.

~X

Represents a character value from 0 to 26. Replace the X like in the following example: ~@=means character ASCII 0, ~A means character ASCII 1, ~B means character 2, ~C means character ASCII 3 and so on. The most used symbols are ~I for a tab and ~M which represents a carriage return.

B.2. Macro 5 and 6

Data matrix provides a way of abbreviating two industry specific header and trailer in one symbol character. This feature aims to reduce the overall symbol size. They must appear at the beginning of the input. You can use tilde codes ~5 and ~6 to enter them.

Table B.1. Macro 5 and 6

Tilde Sequence	Name	Interpretation	
		header	**trailer**
~5	05 Macro	[)>[RS]05[GS]	[RS][EOT]
~6	06 Macro	[)>[RS]06[GS]	[RS][EOT]

B.3. Extended Channel Interpretation (ECI)

ECI allows data streams with different interpretations to be encoded in one symbol. Without ECI the default interpretation is ISO8859-1, the Latin alphabet used in western Europe. Started from version 3.30, the ECI is supported on DataMatrix Fontware.

An ECI value can be any number between 0 and 999999. The tilde code ~7nnnnnn is used to enter the ECI value. The tilde code sequence can appear at any places of the input, but there must be exact 6 digits following ~7. For example, to start ECI interpretation 10, enter ~7000010.

B.4. Structural Append (SA)

The structural append feature allows up to 16 symbols in a structure. A capable reader can either buffer the contents of each symbol until all symbols are read, or deliver the data chunk by chunk.

To encode structural append, you must provide three settings for each symbol:

- *Symbol Sequence Indicator (SI)*. The sequence indicator is 1-based index which identify the position of this particular symbol in the group. Can be any number between 1 and 16.
- *Total number of symbols (TS)*. This value indicates the number of total symbols. Can be any number between 1 and 16. The value should be consistent among all symbols in the group.
- *File Identification Number(FID)*. Identify the symbol group. This number must remain the same among all the symbols in the group.

The tilde codes sequence is expressed in the following format:

```
~2[SI][FID][TS]
```

For example, tilde code sequence `~2[1][126][6]` indicates that the current symbol belongs to a group with file identification number as 126, and there are 6 symbols in total in this group.

The ~2 tilde code sequence must appear at the end of the input. All three fields are required and must be enclosed with square brackets ([and]) and must follow the tilde code ~2.

B.4.1. File ID (FID)

The File ID is a number remaining constant among all symbols in a group. It uniquely identifies the symbol group. The value for this field should be between 1 and 64516.

B.4.2. Sequence Indicator (SI)

Sequence indicator is a 1-based index number of the current symbol. In a group with total 10 symbols, the first symbol has the SI as 1 and the last has the SI as 10.

B.4.3. Total Number of Symbols

The Total number of symbols indicates how many symbols in the group.

Appendix C. Size ID Parameter (updated in version 5.1)

The encoder is enhanced with the release of version 5.1. The change enables user to specify encoding modes and specify if the encoder increases size automatically if it determines that the data cannot be encoded with the size specified. No new API is introduced. Instead, the new features are retrofitted into the size ID parameter.

Prior to version 5.1, the size ID parameter allowed is an integer between 0 and 29 which corresponds to 30 datamatrix sizes, or -1/-2 for automatic size selection. In version 5.1, we added option to allow caller application to set additional bits to tune encoder behavior.

An integer on Windows platform is at least 32 bits. The bit numbering starts at zero for the least significant bit (LSB 0). Only 16 bits are used (LSB 15 to LSB 0). Other bits should be set to 0 for the future use.

In order to be backward compatible with existing applications, value -1 and -2 are handled separately. If the two values are encountered, the logic follows the earlier version - automatic size selection and automatic encodation mode selection.

Secondly, the 16 bits are divided into 2 bytes. The first byte, containing LSB 15 to 8, is called modifier byte. The modifier byte specifies additional behaviors of the encoder. The second bye, LSB 7 to 0, specifies the datamatrix size ID.

C.1. Modifier byte

The 8 bits in the modifier byte are referred bit 7 to bit 0. The below lists the meaning of each bit:

Bit 7: if this bit is set, the encoder should report an error instead of increasing the internal data matrix size if it finds that the size specified cannot encode the data.

Bit 6-4: reserved for the future use and must be set to 0.

Bit 3-0: the value of the four bits determines the encoding mode selection.

- Value 0 indicates that encoder selects the encodation mode automatically, with the goal to minimize the size required. This the behavior defined in the ISO standard.
- Value 1 indicates that encoder must use C40 encodation mode throughout. If characters that cannot be encoded by C40 is found, the encoder reports an error.
- Value 2. Same as value 1, except the encodation mode used is X12.
- Value 3. Same as value 1, except the encodation mode used is TEXT.
- Value 4. Same as value 1, except the encodation mode used is EDIFACT.
- Value 5. Same as value 1, except the encodation mode used is ASCII.
- Value 6. Same as value 1, except the encodation mode used is BASE256.
- Value 7. Reserved for the future
- Value 8 indicates that encoder selects the encodation mode based on Royal Mail MailMark® specification - i.e. use C40 for the first 45 bytes and encode the remaining bytes in automatic encodation selection.

C.2. Size ID byte

The value of this byte corresponds to datamatrix size. A data matrix symbol may have 30 different sizes, numbered from 0 to 29. Two values, 30 and 31 are supported for automatic size selection behavior.

- Value 0-29: the requested datamatrix size. If the bit 7 of the modifier byte is set, the encoder reports an error if the size specified is too small to encode the data. Otherwise, the encoder increases the size automatically.
- Value 30: this value means to select the size automatically, started with rectangular shape. The bit 7 of the modifier byte is ignored.
- Value 31: this value means to select the size automatically, started with square shape. The bit 7 of the modifier byte is ignored.

Note that value 30 and 31 behavior is different from -1 and -2: value -1 and -2 also indicate automatic encodation mode selection. For value 30 and 31, the encodation mode selection depends on the modifier byte.

C.3. Royal Mail MailMark Barcode

The UK Royal Mail uses datamatrix for their MailMark barcodes. However, instead of allowing automatic encodation mode selection, the MailMark type 7, type 9 and type 29 require the first 45 bytes to be encoded in C40 encodation mode. Type 12 barcode requires all the data to be encoded in C40 mode. Prior to 5.1, our encoder is set to automatic mode selection and therefore produces datamatrix barcodes that are more compact but not compatible with the requirements.

With the release of version 5.1, users can create MailMark barcodes with size ID parameter set according to the table below.

Table C.1. Size ID value for MailMark barcodes

Barcode Type	Size ID	Comment
2D CMDM type 7	0x8807(hex), 34823(decimal)	CMDM type 7 corresponds to datamatrix size 7 (24x24 modules). CMDM size is fixed therefore bit 7 is set (0x80). Bit 4-0 has a value of 8 for mailmark C40 mode selection.
2D CMDM type 9	0x8809(hex), 34825(decimal)	CMDM type 9 corresponds to datamatrix size 9 (32x32 modules). CMDM size is fixed therefore bit 7 is set (0x80). Bit 4-0 has a value of 8 for mailmark C40 mode selection.
2D CMDM type 29	0x881D(hex), 34845(decimal)	CMDM type 7 corresponds to datamatrix size 29 (16x48 modules). CMDM size is fixed therefore bit 7 is set (0x80). Bit 4-0 has a value of 8 for mailmark C40 mode selection. The lowest byte has a value of 29 (1D in hex)
2D CMDM type 12	0x810C(hex), 33036(decimal)	CMDM type 7 corresponds to datamatrix size 12 (44x44 modules). CMDM size is fixed therefore bit 7 is set (0x80). Note that type12 requires all data to be encoded in C40 therefore bit 4-0 has a value of 1. The value of the lowest byte is data matrix size ID, 12 (0C in hex)

Appendix D. Unicode String Encode Support

Data matrix barcode (as defined in ISO 16022) does not support Unicode natively. The default character set is ISO8859-1. Theoretically the support could become available using ECI; however there is no ECI published for any of Unicode character set (UTF16, UTF8 etc.). There is also lack of ECI support in most 2D barcode readers.

Because there is a demand on encoding characters outside ISO8859-1, several methods have been developed. The common approach is to encode characters in native character set, and the reader is configured to read based on the default locale. This approach produces the smallest barcode as possible with one major caveat. The same data matrix code is decoded into different text when read by readers with different locale configured. In many use cases this is not an issue, as a 2D barcode with Chinese text encoded is intended to be used in China only.

A new API DataMatrixEncode2W is added to accept a UTF16 string. Internally, the encoder examine the contents of the UTF-16 string. If all characters fall into ISO8859-1, it converts them into ISO8859-1 and encoded as is. Otherwise, it converts the UTF16 string into UTF-8 with BOM, and encode the result. You can still use the DataMatrixEncode2 API and take care of the character set conversion by yourself, for cases that you are required to use local character set.

Several components that accept Unicode parameters are updated in 5.1 release. If you are working exclusively with ASCII or ISO8859-1, you wont' see any changes in the results. Previously, characters outside ISO8859-1 are converted to its ANSI counterpart with the default locale. Now with 5.1 release, the whole string will be converted to UTF-8 with BOM. This makes the data matrix code portable among countries. Those components include DataMatrix ActiveX control, GUI encoder and Crystal Reports UFL.

Appendix E. Fontware License Agreement

By using or installing font software (referred as "Fontware" and "SOFTWARE" in this agreement, including fonts, components, source code, install program etc.) created by Morovia Corporation (referred as "Morovia" below), you (or you on behalf of your employer) are agreeing to be bound by the terms and conditions of this License Agreement. This License Agreement constitutes the complete agreement between you and Morovia. If you do not agree to the terms and condition of the agreement, discontinue use of the Fontware immediately.

License Grant

Number of Installation or Users: In consideration for the license fee paid, Morovia grants to you only, the License, the non-exclusive, non-transferable right to use the font in accordance with the license you purchase. If you are using this product for your employer, this agreement also includes your employer. You may only use the font on computers (CPUs) for which the Fontware is licensed.

The Single User License allows an individual to use the license Fontware on 1 CPU in your organization connected to any number of printers or other image producing devices. If you install or use the fonts on more than one CPU in your organization, or multiple individuals access the barcode printing functionality (e.g. printing from a printer server), multiple single user licenses must be purchased.

The Corporate License allows unlimited use of the licensed fonts in the organization that purchases it. After the Corporate License is purchased, the fonts may be installed and used on multiple CPUs in that organization without any additional fees to be paid to Morovia.

The Developer License allows 1 developer to install the fonts within his organization (Corporate License) as well as rent, lease or distribute the licensed fonts bundled with an application up to 10,000 users (formerly referred as Distribution License). The developer may not resell, rent, lease or distribute the fonts alone, they must be bundled with an application or with the application installation files. The developer may not resell, rent, lease or distribute the fonts in any way that would directly compete with Morovia. If use exceeds 10,000 concurrent users or installations, additional Developer License is required.

Copyright

All title and intellectual property rights in and to the Software Product (including but not limited to any images, photographs, animations, video, audio, music, text, and "applets" incorporated into the Software Product), the accompanying printed materials, and any copies of the Software Product are owned by Morovia. All title and intellectual property rights in and to the content that is not contained in the Software Product, but may be accessed through use of the Software Product, is the property of the respective content owners and may be protected by applicable copyright or other intellectual property laws and treaties. This Agreement grants you no rights to use such content. If this Software Product contains documentation that is provided only in electronic form, you may print one copy of such electronic documentation. You may not copy the printed materials accompanying the Software Product.

Distribution Limits

You must own a Developer License to distribute fonts outside your organization. You are allowed to distribute the software inside or outside your organization for up to 10,000 copies. When you distribute the software, you adhere to the following terms: (a) You may not resell, rent, lease or distribute the Software alone. The Software must be distributed as a component of an application and bundled with an application or with the application's installation files. The Software may only be used as part of, and in connection with, the bundled application. (b) You may not resell, rent, lease or distribute Software in any way that would compete with Morovia. (c) You must include the following MOROVIA copyright notice in your Developed Software documentation and/or in the "About Box" of your Developed Software, and wherever the copyright rights notice is located in the Developed Software ("Portions Copyright (c) Morovia Corporation 2004. All Rights

Reserved."). (d) You agree to indemnify, hold harmless, and defend MOROVIA, its suppliers and resellers, from and against any claims or lawsuits, including attorney's fees that may arise from the use or distribution of your Developed Software. (e) you may use the SOFTWARE only to create Developed Software that is significantly different than the SOFTWARE. (f) You must use font embedding technology when you use the SOFTWARE in PDF and WORD documents; (g) You must specify the exact domain name when creating embedded fonts for web pages. (h) All GUI programs coming with the font package are non-distributable.

Termination

This Agreement is effective until terminated. This Agreement will terminate automatically without notice from Morovia if you fail to comply with any provision contained here. Upon termination, you must destroy the written materials, the Morovia product, and all copies of them, in part and in whole, including modified copies, if any.

Warranty & Risks

The fonts provided by Morovia are licensed to you as is and without warranties as to performance of merchantability, fitness for a particular purpose or any other warranties whether expressed or implied. You, your organization and all users of the font, assume all risks when using it. Morovia shall not be liable for any consequential, incidental, or special damages arising out of the use of or inability to use the font or the provision of or failure to provide support services, even if we have been advised of the possibility of such damages. In any case, the entire liability under any provision of this agreement shall be limited to the greater of the amount actually paid by you for the font or US $5.00.

Glossary

DLL

Acronym for Dynamic Link Library, a library of executable functions or data that can be used by a Windows application. A DLL can be used by several applications at the same time. Some DLLs are provided with the Windows operating system and available for any Windows application. Other DLLs are written for a particular application and are loaded with the application.

EMF

Acronym for Enhanced MetaFile. A newer 32-bit version of Windows MetaFile. EMF contains frame information and contain more drawing commands then its predecessor, WMF.

PNG

Acronym for Portable Network Graphics. PNG is a bitmap image format that employs lossless data compression.

RTF

Acronym for Rich Text Format. A document file developed by Microsoft since 1987 for cross-platform document interchange. Most word processors are able to read and write RTF documents.

Index

A

Adding DataMatrix symbols to SQL Server Reporting Service, 37
Adding Datamatrix to Crystal Reports, 27
Advanced features
  ECI, 44
  Macro 5, 44
  Macro 6, 44
  SA, 44

D

data matrix, 3
  data capacity, 3
  ECC200, 3
DataMatrix Control
  BackColor Property, 22
  CopyToClipboard Method, 24
  ExportImageRaster Method, 24
  ExportImageVector Method, 24
  ForeColor Property, 22
  GetActualSizeID Method, 25
  ModuleWidth Property, 22
  Picture Property, 22
  SizeID Property, 23
  Specification, 21
  TargetDPI Property, 23
  Text Property, 23

E

Extended Channel Interpretation (see Advanced features)

I

Input format, 43

S

Size ID Parameter, 47
Structural Append (see Advanced features, SA)

T

Technical Support, 41
Tilde codes, 43