

2D Barcode Fonts

PDF417 Barcode

<http://barcoderesource.com/pdf417barcode.shtml>

Copyright (c) 2009-2022, ConnectCode

All Rights Reserved.

ConnectCode accepts no responsibility for any adverse affect that may result from undertaking our training.

Microsoft and Microsoft Excel are registered trademarks of Microsoft Corporation. All other product names are trademarks, registered trademarks, or service marks of their respective owners

Table of Contents

1.	PDF417 Barcode.....	1-1
1.1	PDF417 Barcode.....	1-1
1.2	Error Correction.....	1-1
1.3	Data Compaction	1-1
1.3.1	Auto Data Compaction	1-1
1.4	Parameters of the PDF417 Barcode.....	1-2
1.4.1	Row Height.....	1-2
1.4.2	Number of Columns	1-2
1.4.3	Error Correction Level	1-2
1.4.4	Truncate	1-3
1.5	PDF417 Barcode Fonts	1-3
2.	Font Encoder	2-4
3.	Component Object Model Library	3-5
3.1	Tutorial on creating PDF417 using COM	3-6
4.	PowerBuilder	4-10
4.1	Tutorial on creating a PDF417 in PowerBuilder.....	4-10
5.	.NET Standard SDK.....	5-16
5.1	.NET Standard PDF417 Library	5-16
6.	.NET SDK	6-17
6.1	.NET Framework 4.0 Notes	6-18
7.	Integrating with Crystal Reports	7-19
7.1	Tutorial for creating the PDF417 Barcode in Crystal Reports	7-19
8.	Access/Excel VBA Integration.....	8-22
8.1	Importing the VB macros to Access.....	8-22
8.1.1	Notes on using the PDF417 Barcode Font in Microsoft Access	8-22
8.2	Importing the VB macros to Excel.....	8-22
8.3	VBA Function Description	8-23

1. PDF417 Barcode

1.1 PDF417 Barcode

PDF417 barcode is one of the earliest and most widely used 2-dimensional barcodes. It is a stacked barcode composed of rows of linear barcodes. Being a 2-dimensional barcode allows it to carry more information than the 1-dimensional barcodes.

“PDF” stands for Portable Data File. The “417” derives from the fact that each encoded codeword in the barcode is represented by four bars and four spaces. The total width of each codeword is 17-module wide and thus the “17” in the name.



1.2 Error Correction

Besides being able to pack more data, the PDF417 barcode provides additional redundancy by using a correction technique known as the Reed Solomon error correction. This allows the barcode to be partially damaged without causing any loss of data. Under the industry specifications, a total of 9 levels of error corrections are supported. The higher the level of error correction, the more redundancy the barcode has. However, it also means that more areas from the barcode will need to be used for error correction and resulting in less data to be encoded into the barcode.

1.3 Data Compaction

The PDF417 barcode supports the following compaction mode. Different compaction mode allows different types of data to be encoded optimally into the barcode.

- Binary - Data Bytes. A maximum of 1.2 bytes per codeword.
- Text - Text Characters e.g. ASCII 32 to ASCII 127. A maximum of 2 characters per codeword.
- Numbers - Digits from 0..9. A maximum of 2.9 digits per codeword.

1.3.1 Auto Data Compaction

ConnectCode provides automatic compaction based on the above compaction mode according to the AIMS industry specifications. The algorithm used first scans the data and decides automatically which compaction mode to use. On top of that, the data can sometimes consist of a combination of binary data, text, and numbers data. In this case, the algorithm will automatically switch between the different modes to provide maximum compaction.

1.4 Parameters of the PDF417 Barcode

The following list the different configurable parameters of the PDF417 barcode. If you are new to this barcode, it is recommended that you use the default or automatic settings mentioned below.

1.4.1 Row Height

The Row Height of the PDF417 Barcode is specified as a multiple over the column width. For example, "3x" means the row is 3 times the height of the column width. This specification of row height is according to the industry specifications.

- 3x - This is the default row height supported. It is also the most commonly used row height.
- 4x - The row height is 4 times the column width.

1.4.2 Number of Columns

A PDF417 barcode will typically expand vertically by adding rows to accommodate more data. However, the barcode can also be expanded horizontally by expanding the number of columns. The diagram below shows a PDF417 barcode with 1 column. You probably see 3 columns below, but it is considered 1 data column.



The PDF417 barcode encoded with the same data but with 2 columns is shown in the diagram below.



Supported Columns

- 1..30
- Auto - This is the recommended setting if you have not used the PDF417 barcode before. The ConnectCode PDF417 Barcode will automatically detect the optimal columns according to industry specifications.

1.4.3 Error Correction Level

The list below shows all the error correction levels supported by the PDF417 barcode. A higher error correction level will mean a more robust barcode that can withstand more damage. However, it will also mean that less data can be encoded into the barcode.

- 0 - Support 2 error codewords. This is the minimum Error Correction Level.
- 1 - Support 4 error codewords.
- 2 - Support 8 error codewords.
- 3 - Support 16 error codewords.
- 4 - Support 32 error codewords.
- 5 - Support 64 error codewords.
- 6 - Support 128 error codewords.
- 7 - Support 256 error codewords.
- 8 - Support 512 error codewords. This is the maximum Error Correction Level.
- Auto - This is the recommended setting if you have not used the PDF417 barcode before. The ConnectCode PDF417 Barcode will automatically detect the optimal Error Level.

1.4.4 Truncate

The right-hand side of the PDF417 barcode can be truncated (removed) without causing any loss of data. This allows the creation of a barcode that takes up a smaller amount of space than a normal PDF417 barcode. The diagram below shows a Truncated PDF417 Barcode.



1.5 PDF417 Barcode Fonts

The following is the list of barcode fonts used by the PDF417 barcode. Different types of fonts have been designed for different types of applications. This is to achieve the most optimal scanning possible.

Reporting Fonts

Font Name	Description	Recommended Sizes	Type of Applications
CCodePDF417_S3 (CCodePDF417_S3_Trial for the Trial version)	PDF417 Barcode with a row height of 3x the column width.	Font Size 14..20	Crystal Reports , other reporting tools, and Microsoft Word 2000.
CCodePDF417_S4 (CCodePDF417_S4_Trial for the Trial version)	PDF417 Barcode with a row height of 4x the column width.		

Application Integration Fonts

Font Name	Description	Recommended Sizes	Type of Applications
CCodePDF417_S3X (CCodePDF417_S3X_Trial for the Trial version)	PDF417 Barcode with a row height of 3x the column width.	Font Size 2..10 (Half point size such as 2.5 is also supported)	Wordpad, Microsoft Word 2003 - 201 (or onwards), PowerBuilder, and Visual Studio .NET Applications
CCodePDF417_S4X (CCodePDF417_S4X_Trial for the Trial version)	PDF417 Barcode with a row height of 4x the column width.		

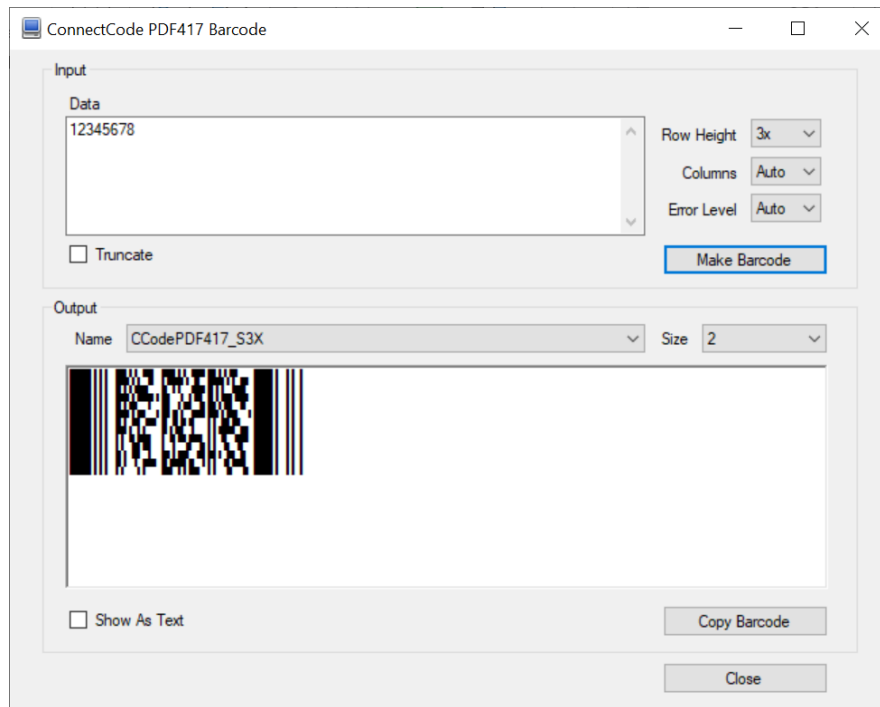
Note

1. You may see spaces between multiple Rows when you use the PDF417 barcode fonts in Microsoft Excel. This is due to the way Microsoft Excel handles multiple lines of text in a cell. The spaces can be easily removed by copying the resulting output, or barcode, into Microsoft Word before printing them. Please note that it is not possible to remove the spaces within Excel. This is a problem inherent in all barcode font products.
2. You may not see your application or the tools listed above. It does not mean the barcode fonts cannot be used by your project/product. You can try out the two fonts above and will most likely be able to find a suitable one. Alternatively, you can contact us for suggestions.

2. Font Encoder

The PDF417 Barcode Font package in ConnectCode comes bundled with a Font Encoder that allows you to encode the barcode quickly and easily. This is useful if you would like to encode a single barcode to be pasted into your brochure, packaging, or product items. The Encoder supports all the different parameters as described in the sections above.

The Row Height, Columns, Truncate, and Error Level are parameters that have been described in the previous section and thus will not be described again here.



The screenshot shows a window titled "ConnectCode PDF417 Barcode". It is divided into two main sections: "Input" and "Output".

Input Section:

- Data:** A text box containing the number "12345678".
- Row Height:** A dropdown menu set to "3x".
- Columns:** A dropdown menu set to "Auto".
- Error Level:** A dropdown menu set to "Auto".
- Truncate:** An unchecked checkbox.
- Make Barcode:** A blue button.

Output Section:

- Name:** A dropdown menu set to "CCodePDF417_S3X".
- Size:** A dropdown menu set to "2".
- Barcode:** A large rectangular area displaying a generated PDF417 barcode.
- Show As Text:** An unchecked checkbox.
- Copy Barcode:** A button.
- Close:** A button.

The Font Name and Font Size in the "Output" section can be changed after the PDF417 barcode is created. This allows the height and the size of the barcode to be changed after the barcode is created.

The "Show As Text" option allows you to see the text output of the barcode in a normal text font. The "Copy Barcode" button allows the barcode to be copied onto the clipboard of Microsoft Windows. This allows you to paste the barcode onto other applications easily.

3. Component Object Model Library

This tutorial illustrates the use of a COM (Component Object Model) object library with a True Type Font (PDF417 Barcode Font), provided in ConnectCode PDF417 package, to create an ISO/IEC 15438:2015 standard-compliant PDF417 in a .NET Windows Form application.

Prerequisites

- ConnectCode [PDF417](#) package is installed
- PDF417COMLibrary.dll in the Resource\PDF417COMLibrary subdirectory of ConnectCode PDF417 package. The PDF417 class library has been compiled with the "Register for COM interop" Visual Studio project property, exposing a COM-callable wrapper that enables COM interaction.
- Visual Studio 2015/2017/2019/2022
- Administrator Rights

3.1 Tutorial on creating PDF417 using COM

1. Launch the Visual Studio Developer Command Prompt as Administrator from the Windows Start Menu.
2. In the Developer Command Prompt, use the "cd" command to go to the PDF417 COM Library folder.

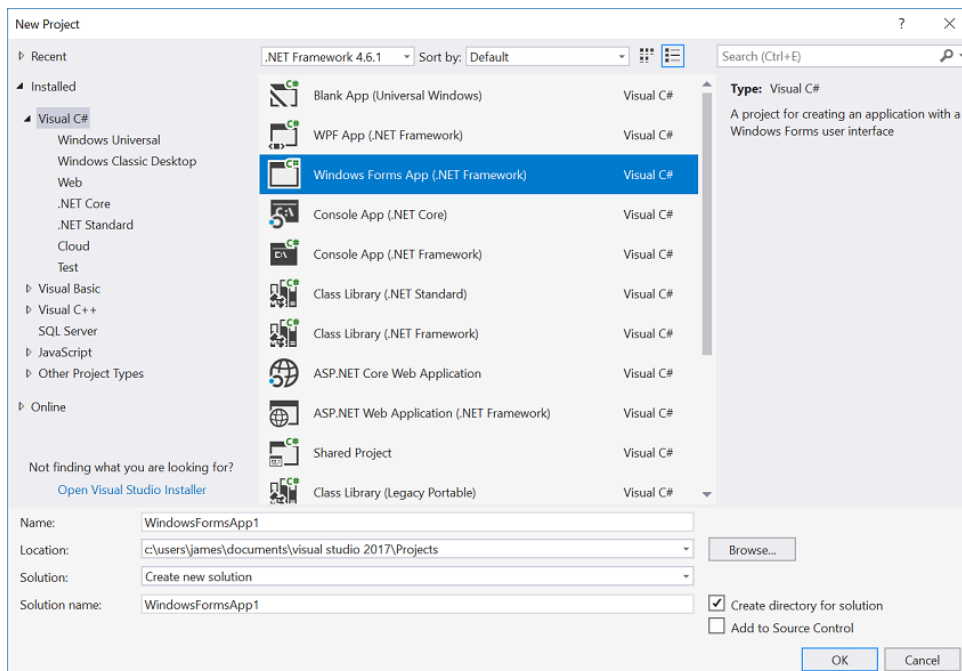
```
cd C:\Program Files (x86)\ConnectCodePDF417\Resource\PDF417COMLibrary  
(or ConnectCodePDF417Trial if you are using the Trial package)
```

3. Enter the following command in the Developer Command Prompt to use Regasm.exe to register the PDF417COMLibrary assembly for use with COM.

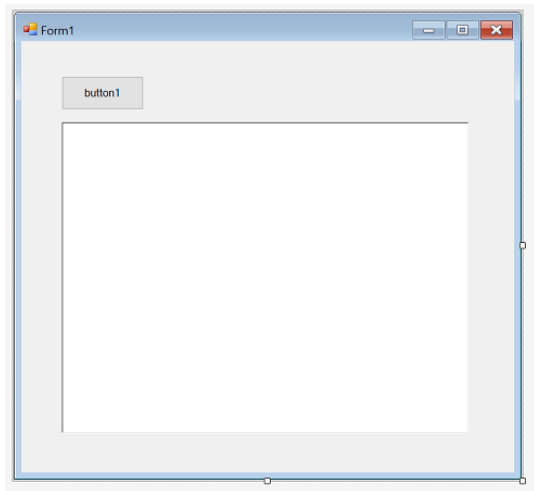
```
Regasm PDF417COMLibrary.dll /tlb:PDF417COMLibrary.tlb /codebase
```

Regasm.exe adds information about the class to the system registry so that COM clients can use the .NET Framework class transparently. The "tlb" option generates a type library defined within the assembly.

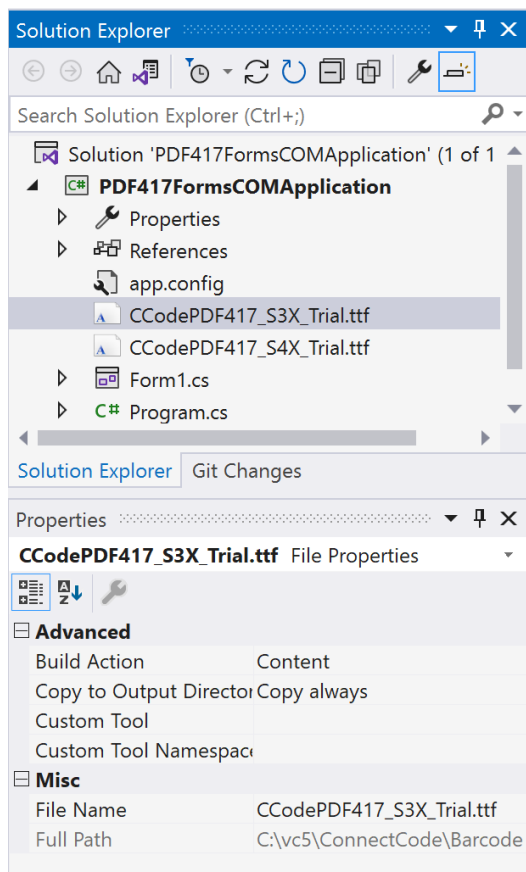
4. Launch Visual Studio. Create a new Windows Form project by clicking on "File->New Project", selecting a "Windows Forms App" and clicking on "Create" button.



5. Double click on "Form1.cs" in the "Solution Explorer" and add a "Button" and a "RichTextBox" from the Visual Studio Toolbox. You should see your Windows Form like the screenshot below.



6. Right-click on the "WindowsFormApp1" project in the "Solution Explorer" and select "Add->Existing Item". Navigate to the "C:\Program Files (x86)\ConnectCodePDF417\" folder and select the "CCodePDF417_S3X.ttf" font. If you are using the trial version, select the " CCodePDF417_S3X_Trial.ttf" font instead.



7. In "Solution Explorer", select the " CCodePDF417_S3X.ttf" font and change the "Build Action" to "Content" and "Copy to Output Directory" to "Copy Always" in the Properties pane. This will ensure that Visual Studio deploys the PDF417 barcode font for use with the Windows Form application.

8. Double click on "Form1.cs" in the "Solution Explorer". In the designer, double click on the Button. This will generate the button1_Click function in the editor. Enter the C# programming codes as shown below:

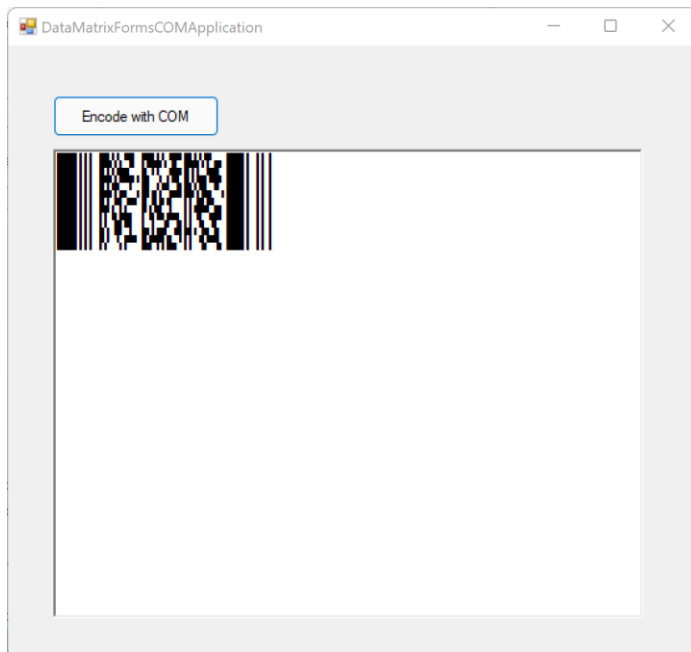
```
using System.Drawing.Text;
.
.
.
private void button1_Click(object sender, EventArgs e)
{
    try
    {
        PrivateFontCollection pfc = new PrivateFontCollection();
        pfc.AddFontFile("CCodePDF417_S3X_Trial.ttf");
        //pfc.AddFontFile("CCodePDF417_S4X_Trial.ttf");
        richTextBox1.Font = new Font(pfc.Families[0], 1, FontStyle.Regular);

        Type comObjectType =
            Type.GetTypeFromProgID("Net.ConnectCode.PDF417COMLibrary");
        dynamic theComObject = Activator.CreateInstance(comObjectType, false);

        //or
        //Guid myGuid = new Guid("7444CF44-564C-4DE9-B3B6-8B769B7C2549");
        //Type comObjectType = Type.GetTypeFromCLSID(myGuid);
        //dynamic theComObject = Activator.CreateInstance(comObjectType, false);

        string inputData = "12345678";
        int cols = 0; //Auto
        int errorLevel = -1; //Auto
        int truncate = 0;
        string result1 = theComObject.Encode_PDF417(inputData,cols,errorLevel,truncate);
        string result = result1;
        richTextBox1.Text = result;
        System.Diagnostics.Debug.WriteLine(result1);
    }
    catch (Exception ex)
    {
        System.Diagnostics.Debug.WriteLine(ex);
    }
}
```

The C# function above creates a PDF417 COM object and then uses it to generate a PDF417 barcode with the input data "12345678". The result is placed in "richTextBox1" and the final output is displayed with the CCodePDF417_S3X True Type font. When you run the application, click on the "Encode with COM" button, you should see the PDF417 barcode as shown below.



The "PDF417COMApplication" folder in "C:\Program Files (x86)\ConnectCodePDF417\Resource" contains the full source code of the above application.

4. PowerBuilder

This tutorial illustrates the use of a COM (Component Object Model) library and a barcode font available in ConnectCode PDF417 package for creating PDF417 barcode in PowerBuilder. The generated PDF417 complies with the ISO/IEC 15438:2015 standards and can meet the strictest requirements of the Auto-ID industry.

Prerequisites

- PowerBuilder v12 (or APPEON PowerBuilder 2017-2021. In 2016, SAP and Appeon have entered into an agreement whereby Appeon would be responsible for developing and marketing PowerBuilder.)
- ConnectCode PDF417 package is installed

4.1 Tutorial on creating a PDF417 in PowerBuilder

1. Launch a Windows Command Prompt as Administrator. In the Command Prompt, enter the following command to go to the PDF417COMLibrary folder.

```
cd C:\Program Files(x86)\ConnectCodePDF417\Resource\PDF417COMLibrary
```

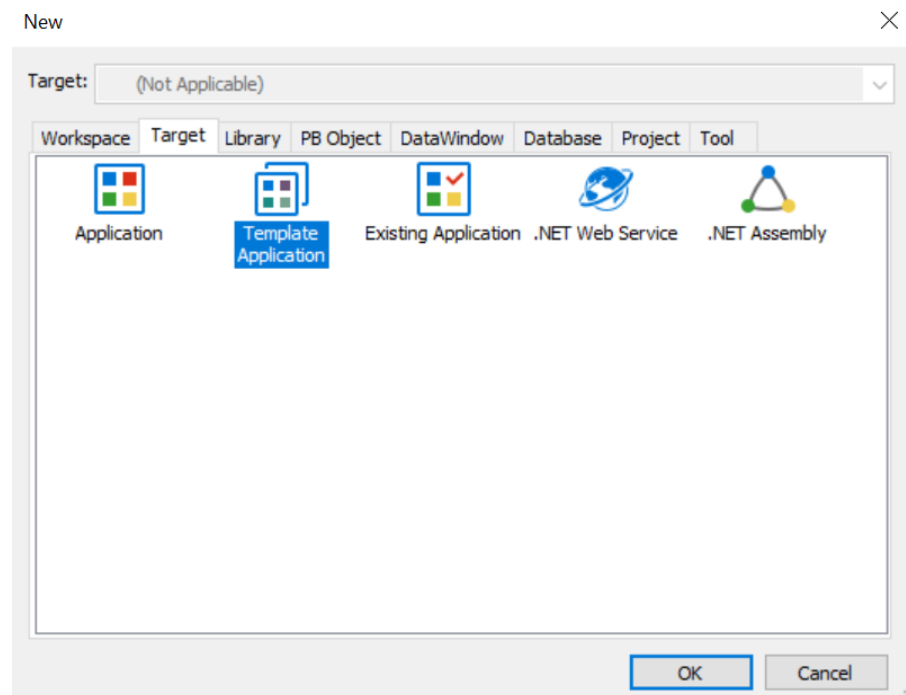
2. Next, use the Assembly Registration Tool (Regasm.exe) to register the PDF417COMLibrary.dll assembly with COM.

```
Regasm PDF417COMLibrary.dll /tlb:PDF417COMLibrary.tlb /codebase
```

If Regasm is not available, you can verify if the following folder exists and add the folder into your PATH. The folder may be different depending on your version of .NET.

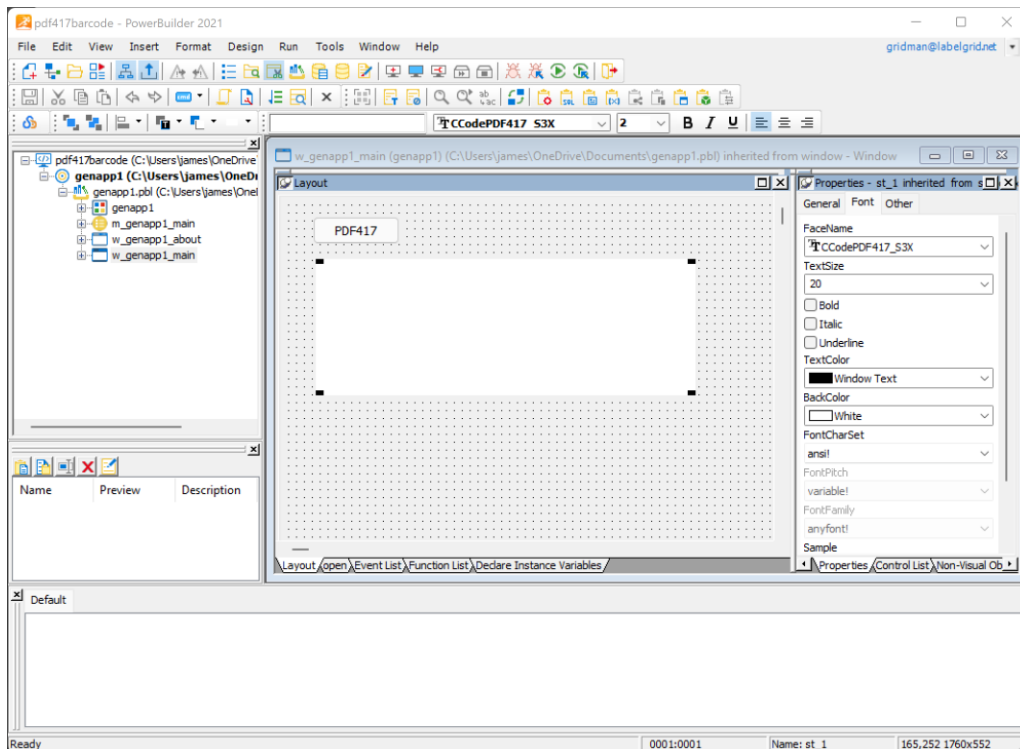
```
C:\Windows\Microsoft.NET\Framework64\v4.0.30319\
```

3. Launch "PowerBuilder" and create a new "Template Application" in the "Target tab".



4. You can select "SDI application" as the "Application Type" and "None" in Connectivity Options" to create a sample application.

5. Click on "w_genapp_main" when the application is created. From the menu select "Insert->Control->CommandButton" and rename the button to "PDF417". Next, insert a "StaticText" control, set its background to "White", and layout the components as shown in the screenshot below.



A PDF417 barcode (ISO/IEC 15438:2015) can have a row height that is 3x or 4x to the column width.

- CCodePDF417_S3X - PDF417 Barcode Font with a row height of 3x the column width.
- CCodePDF417_S4X - PDF417 Barcode Font with a row height of 4x the column width.

Select the "StaticText" control, change the Font to **CCodePDF417_S3X** (or CCodePDF417_S3X_Trial) and the Font Size to 2 to fit the barcode nicely on the "StaticText" control. The output generated by the PDF417COMLibrary will be placed in the "StaticText" control and displayed as a barcode with the CCodePDF417_S3X True Type font.

6. Next, double click on the button and add the following script:

```
OLEObject barcode
int return_code
barcode = CREATE OLEObject
return_code=barcode.ConnectToNewObject("Net.ConnectCode.PDF417COMLibrary")
if return_code<>0 then
    destroy barcode
    messagebox ("Error","PDF417COMLibrary not available")
else
    string result
    string input="12345678"
    int cols = 0
    int errorLevel = -1
    int truncate = 0
    result=barcode.Encode_PDF417(input,cols,errorLevel,truncate)
    st_1.Text=result
    destroy barcode
end if
```

In the source code above, an OLE object is created with the PDF417COMLibrary. The "Encode_PDF417" method is used to generate the barcode.

First Parameter: Input String

Second Parameter: Columns

- 0 for Auto
- 1 to 30 Columns

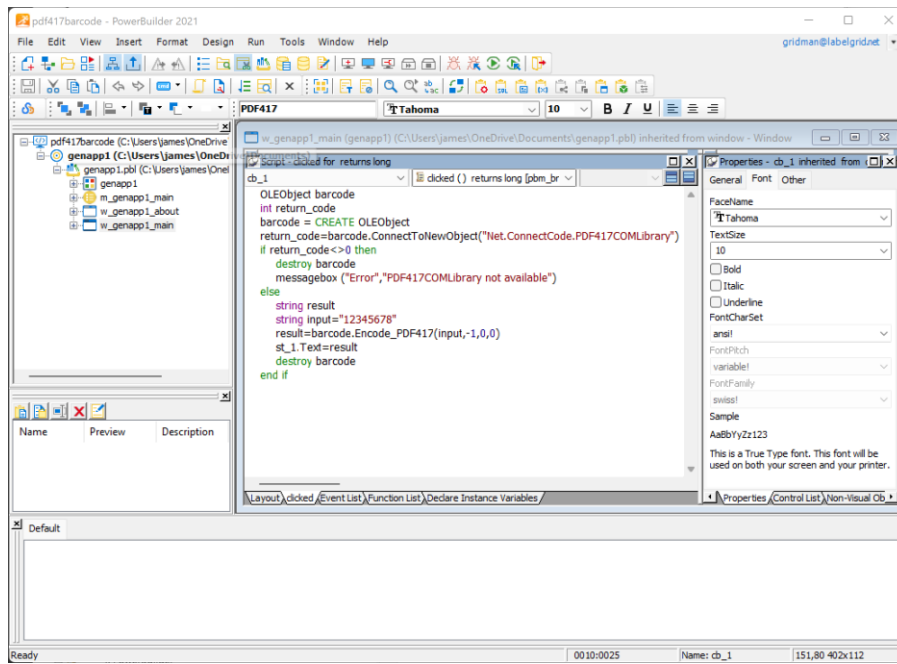
Third Parameter: Error Level

- -1 - Auto Detect Optimal Error Level (Recommended)
- 0 - Support 2 error codewords. This is the minimum Error Correction Level.
- 1 - Support 4 error codewords.
- 2 - Support 8 error codewords.
- 3 - Support 16 error codewords.
- 4 - Support 32 error codewords.
- 5 - Support 64 error codewords.
- 6 - Support 128 error codewords.
- 7 - Support 256 error codewords.
- 8 - Support 512 error codewords. This is the maximum Error Correction Level.

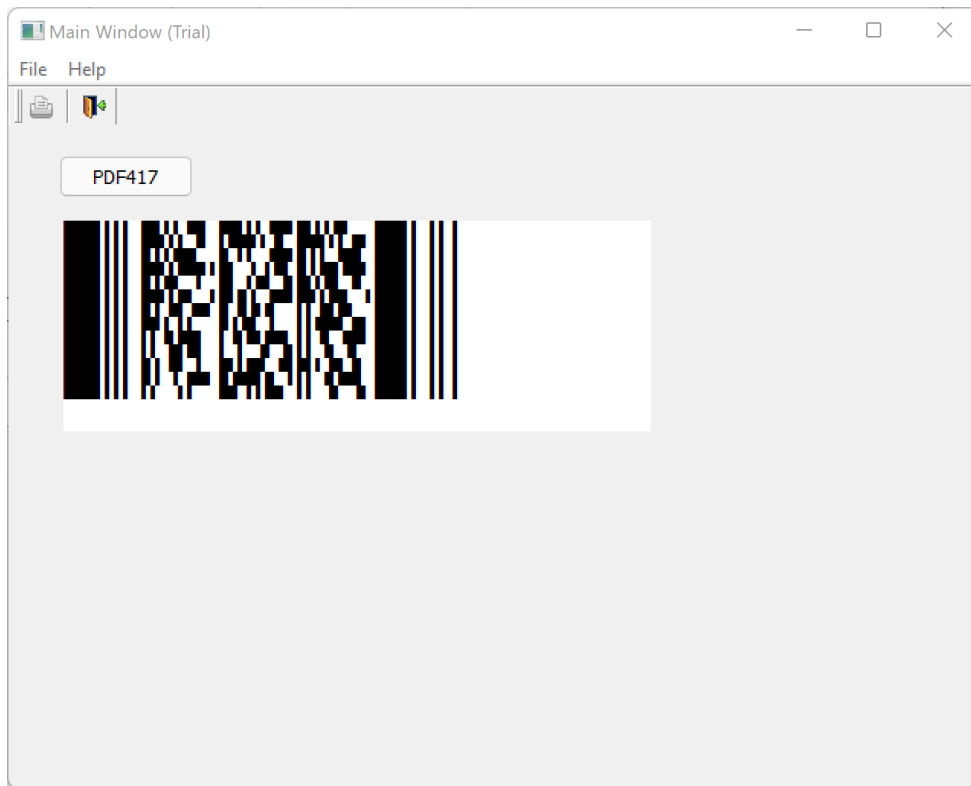
Fourth Parameter: Truncate

The right-hand side of the PDF417 barcode can be truncated (removed) without causing any loss of data. This allows the creation of a barcode that takes up a smaller amount of space than a normal PDF417 barcode. The diagram below shows a Truncated PDF417 Barcode.





7. Run the application by going to the menu and select "Run->Select and Run". Click on the "PDF417" button and see that you get the PDF417barcode output in the "StaticText" control. If you get an error message saying that " PDF417COMLibrary is not available", check that you have carried step 2 successfully.



5. .NET Standard SDK

.NET Framework is a software framework that is developed by Microsoft to run primarily on Microsoft Windows. Over the years, the framework has been forked and enhanced to serve many different purposes. For example, the Universal Windows Platform uses a specific set of APIs from the .NET Framework to help programmers develop apps for the Windows Store. The .NET Core Framework also uses a subset of APIs from the .NET Framework to support the development of applications that can run on different operating systems such as Windows, Mac, and Linux.

.NET Standard is a specification of common APIs that are available on the different .NET frameworks. By creating a class library (DLL) that targets the .NET Standard, a developer can be assured that his library can be used or shared by projects developed on the various .NET frameworks.

As of .NET Standard v2.0, the .Net Standard specification is implemented by the following frameworks:

- .NET Core
- .NET Framework
- Mono
- Xamarin.iOS
- Xamarin.Android
- Universal Windows Platform
- .NET MAUI
- Windows App SDK

5.1 .NET Standard PDF417 Library

- [Resource/NETStandard/netstandard2.0/NETStandardPDF416.dll](#)

The PDF417 Barcode package includes a .NET Standard compliant PDF 417 class library that targets the .NET Standard 2.0 specification. This library can be used by projects developed on different .NET frameworks and when used together with ConnectCode Barcode Fonts, generates barcodes of the highest quality that can meet the strictest requirements of the auto-id industry.

6. .NET SDK

A .NET Barcode SDK is also bundled in the ConnectCode PDF417 Barcode Font package. This SDK can be bundled in your applications if you have purchased the necessary distribution licenses.

Library Name

PDF417.dll

Namespace

ConnectCode.BarcodeFonts2D

Class Name

PDF417

Requirements

.NET 2.0 and onwards

Constructors and Functions

PDF417(String data, int columns, int errorlevel, int truncate);

This is the constructor for the PDF417 barcode. It is used for initializing the PDF417 barcode.

data : The data input string to be encoded as a barcode.

columns : Number of Columns in the PDF417 barcode. 1..30 or 0 for Automatic

errorlevel : The Error Correction Level. 0..8 or -1 for Automatic

truncate : 0 for non truncated PDF417 barcode. 1 for truncated.

String Encode();

This function will encode the barcode based on the parameters specified in the constructor. The result will be returned as a string.

int LengthExceeded();

The Encode() function might return an empty output string due to invalid inputs or if the length of the data exceeded the length specified by the PDF417 specifications. A call to this function after the Encode() function will allow you to determine whether the data length has exceeded.

int GetColumns();

During the creation of the PDF417 barcode, if the number of columns used is "Automatic", the font library will automatically determine the optimal columns. This function can be called after the Encode() function to return the number of columns that is determined by the library.

int GetErrorLevel();

During the creation of the PDF417 barcode, if the error correction level used is "Automatic", the font library will automatically determine the optimal error correction level. This function can be called after the Encode() function to return the error correction level that is determined by the library.

Sample Usage (C#)

```
Using ConnectCode.BarcodeFonts2D;  
.  
.  
.  
PDF417 barcode = new PDF417("12345678",0,-1,0);  
String result = barcode.Encode();  
Font font = new Font("CCodePDF417_S3", 2);  
richTextBox1.Text = outputstr; //private System.Windows.Forms.RichTextBox richTextBox1;  
richTextBox1.SelectAll();  
richTextBox1.SelectionFont = font;
```

Sample Visual Studio Project

1. Name - ConnectCode Encoder
2. Solution Name - ConnectCode.sln
3. Language - C#
4. Requirements - .Net 2.0 and onwards, Visual Studio 2005 and onwards.

6.1 .NET Framework 4.0 Notes

.NET Framework 4.0 includes and uses CLR 4.0. It does not automatically use its version of the common language runtime to run applications that are built with earlier versions of .NET Framework. This is unlike .Net 2.0-3.5 where the framework uses CLR 2.0 to run applications. Basically, there is no version 3 of the CLR.

Hence, ConnectCode 2D Barcode SDK provides two sets of .Net DLLs for different versions of the .Net Framework as shown below:

For .Net 2.0 to 3.5 please use the DLLs and samples in

- /Resource subdirectory
- /.Net Samples subdirectory

For .Net 4.0 please use the DLLs and samples in

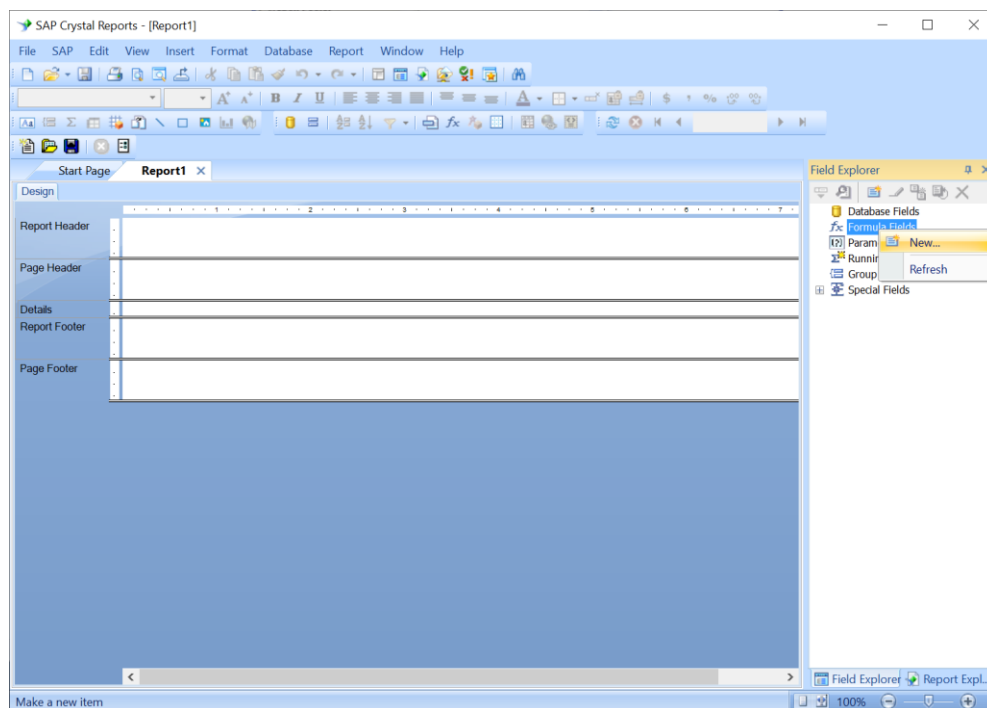
- /Net4 subdirectory
- /Net4/.Net Samples subdirectory

7. Integrating with Crystal Reports

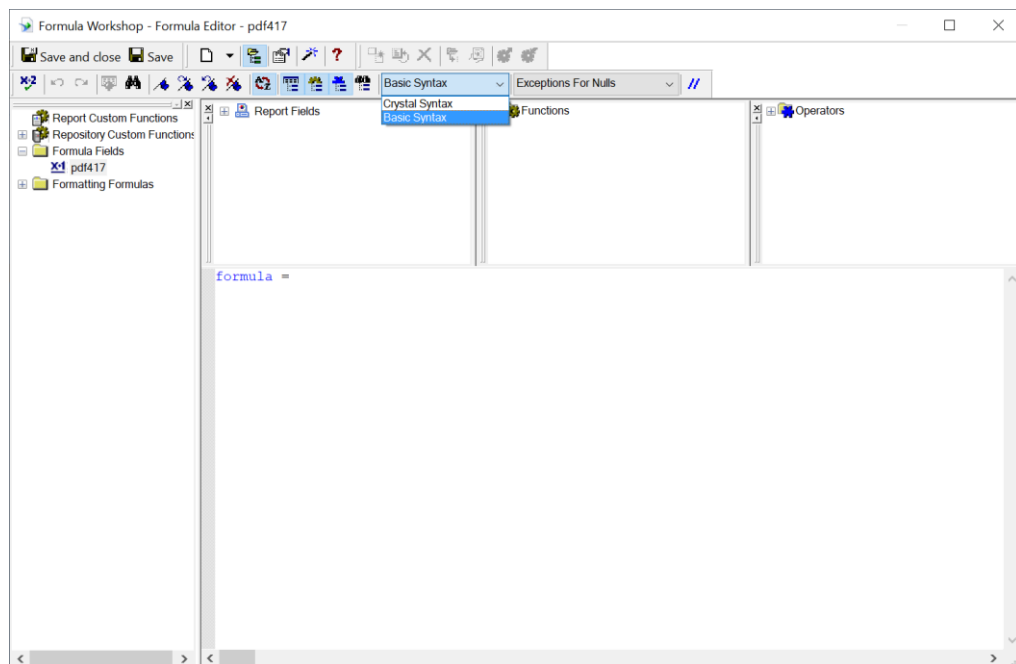
7.1 Tutorial for creating the PDF417 Barcode in Crystal Reports

Prerequisites

- Crystal Reports 2008 – Crystal Reports 2020
 - ConnectCode PDF417 Barcode Fonts package
1. Launch Crystal Reports and create a blank report.
 2. In the Field Explorer, create a new formula by right clicking Formula Field and selecting New. Give the new formula a name (e.g. pdf417).

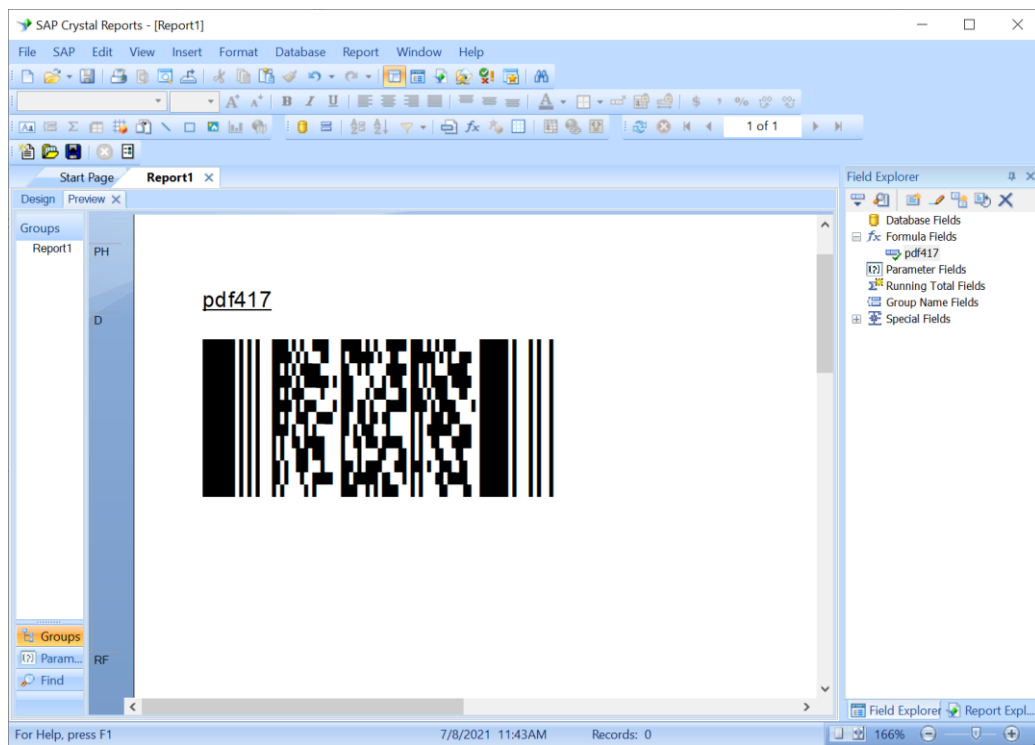


3. In Formula Workshop, set the syntax of the formula to 'Basic Syntax'.



4. Copy the formula for the PDF417 barcode from the CR_Formula.txt file (in the Resource\CrystalReports folder) to the Crystal Report's Formula Editor.
5. Modify the 'data = "12345678"' statement so that it connects to your data source. For example, change the statement to data = { Employee.Name }.
6. Save the formula and drag it to your report to create a formula field. Expand the bounding box of the field to ensure that there is enough space for the PDF417 barcode.
7. Set the font of the formula field to the "CCodePDF417_S3" and font size to 16.

8. Click on Print Preview to preview the PDF417 barcode.



8. Access/Excel VBA Integration

8.1 Importing the VB macros to Access

1. Launch Microsoft Access
2. Click on the menu Tools->Macro->Visual Basic Editor.
3. Click on the menu File->Import File...
4. Select the Encoder.bas file that is bundled with ConnectCode. This file is located in the Resource sub directory of where ConnectCode is installed.
5. Click on Save followed by Close in the Visual Basic Editor

8.1.1 Notes on using the PDF417 Barcode Font in Microsoft Access

In Access, the Rich Text property in the textbox object used to display the PDF417 barcode need to be enabled in order for the barcode to display correctly. To turn on the Rich Text option, please follow the steps below:

1. In Access, select the textbox for the PDF417 barcode in the Design View.
2. Click on Properties->Data tab.
3. In the Text Format property, select Rich Text.

8.2 Importing the VB macros to Excel

1. Launch Microsoft Excel.
2. Click on the menu Tools->Macro->Visual Basic Editor.
3. Click on the menu File->Import File...
4. Select the Encoder.bas file that is bundled with ConnectCode. This file is located in the Resource sub directory where ConnectCode is installed.
5. Click on Save followed by Close in the Visual Basic Editor.

8.3 VBA Function Description

Function Name

Encode_PDF417

Description

This function returns the encoded barcode string for the PDF417 barcode. Invalid characters from the input will be automatically removed.

The font name to use with this barcode is CCodePDF417_S3.

Parameters

Param 1 - String value to be encoded as a barcode.

Param 2 - Number of Columns in the PDF417 barcode. 1..30 or 0 for Automatic.

Param 3 - The Error Correction Level. 0..8 or -1 for Automatic.

Param 3 - 0 for non truncated PDF417 barcode. 1 for truncated.

Important Notes

You may see spaces between multiple Rows when you use the PDF417 barcode fonts in Microsoft Excel. This is due to the way Microsoft Excel handles multiple lines of text in a cell. The spaces can be easily removed by copying the resulting output, or barcode, into Microsoft Word before printing them. Please note that it is not possible to remove the spaces within Excel. This is a problem inherent in all barcode font products.