How to add custom images to PDF document in C# and ByteScout Barcode SDK

The tutorial shows how to add custom images to PDF document in C#

On this page you will learn from code samples for programming in C#.Writing of the code to add custom images to PDF document in C# can be done by developers of any level using ByteScout Barcode SDK. Want to add custom images to PDF document in your C# app? ByteScout Barcode SDK is designed for it. ByteScout Barcode SDK is the robost library (Software Development Kit) that is designed for automatic generation of high-quality barcodes for printing, electronic documents and pdf. All popular barcode types are supported from Code 39 and Code 129 to QR Code, UPC, GS1, GS-128, Datamatrix, PDF417, Maxicode and many others. Provides support for full customization of fonts, colors, output and printing sizes. Special tools are included to verify output quality and printing quality. Can add generated barcode into new or existing documents, images and PDF.

Fast application programming interfaces of ByteScout Barcode SDK for C# plus the instruction and the code below will help you quickly learn how to add custom images to PDF document. Follow the instructions from the scratch to work and copy the C# code. Enjoy writing a code with ready-to-use sample codes in C#.

Download free trial version of ByteScout Barcode SDK from our website with this and other source code samples for C#.

C# - Program.cs

```
using System.Diagnostics;
using System.Drawing;
using Bytescout.BarCode;

// This example demostrates adding of barcode together with some custom images (e.g.
target marks) to PDF document.
// Also shows the manual calculation of barcode size from inches to pixels and
document units (points).
// Task: place Code39 barcode of 2" x 3/8" size at 2" from top-right corner and two
target marks at 1" from top-right and bottom-left corners.

namespace AddBarcodeToPdfDocument
{
    class Program
    {
        static void Main(string[] args)
        {
            // Create Barcode instance and set it up
            Barcode barcode = new Barcode("demo", "demo");
            barcode.Symbology = SymbologyType.Code39;
            barcode.Value = "00090112";
```

```
            barcode.DrawCaption = true;
            barcode.CaptionFont = new Font("Courier", 12f, FontStyle.Bold,
GraphicsUnit.Point);
            barcode.DrawQuietZones = false;
            barcode.ResolutionX = barcode.ResolutionY = 300; // High resolution for
better quality on document scaling and printing.

            // Compute barcode image dimension from inches to pixels at 300 DPI:

            int barHeight = (int) (3f/8*300); // = 3/8" * 300 DPI = height of bars
            int captionHeight = (int)
barcode.CaptionFont.GetHeight(barcode.ResolutionY);
            int captionGap = (int) (1f/8*300); // = 1/8" gap

            int barcodeImageWidth = 2*300; // = 2" * 300 DPI = width of barcode
            int barcodeImageHeight = barHeight + captionGap + captionHeight + 28; //
28 = height of watermark text (painted in Trial version only)

            // Get final barcode image:

            barcode.BarHeight = barHeight;
            barcode.FitInto(barcodeImageWidth, barcodeImageHeight,
UnitOfMeasure.Pixel);
            Image barcodeImage = barcode.GetImage();


            // Arrays of images and points to apply to PDF document
            Image[] images = new Image[3];
            Point[] points = new Point[3];

            // Compute coordinates of barcode image and target marks ar 72 DPI
            // (page size is 8.5" x 11", PDF document resolution is always 72 DPI):

            int x = (int) (8.5f*72 - 2*72 - barcodeImageWidth/300f*72); // = page
width - 2" - barcodeImageWidth at 72 DPI = X coordinate to put the barcode image
            int y = 2*72; // = 2" - Y coordinate to put the barcode image

            // Put barcode image into array
            images[0] = barcodeImage;
            points[0] = new Point(x, y);

            // Load target mark image.
            // TargetMark.png is 1/8" 300 DPI image (38x38 pixels)
            Image targetMarkImage = Image.FromFile("TargetMark.png");

            // Coordinates of top-right target mark
            x = (int)(8.5f*72 - 1*72 - 1f/8*72); // = pageWidth - 1" - target mark
width (1/8") at 72DPI = X coordinate to put the barcode image
            y = 1*72; // = 1" - Y coordinate to put the barcode image

            // Put first target mark image into array
            images[1] = targetMarkImage;
            points[1] = new Point(x, y);

            // Coordinates of bottom-left target mark
            x = 1*72; // = 1"
            y = (int)(11f*72 - 1*72 - 1f/8*72); // = page height - 1" - target mark
height (1/8") at 72DPI

            // Put second target mark image into array
```

```
            images[2] = targetMarkImage;
            points[2] = new Point(x, y);

            // Draw images on all PDF document pages.
            barcode.DrawImagesToPDF("wikipedia.pdf", -1 /*all pages*/, images,
points, "result.pdf");

            // Cleanup
            barcodeImage.Dispose();
            targetMarkImage.Dispose();
            barcode.Dispose();

            // Open the result document in default associated application
            Process.Start("result.pdf");

        }
    }
}
```

FOR MORE INFORMATION AND FREE TRIAL:

[Download Free Trial SDK (on-premise version)]

[Read more about ByteScout Barcode SDK]

[Explore documentation]

[Visit www.ByteScout.com]

or

[Get Your Free API Key for www.PDF.co Web API]