



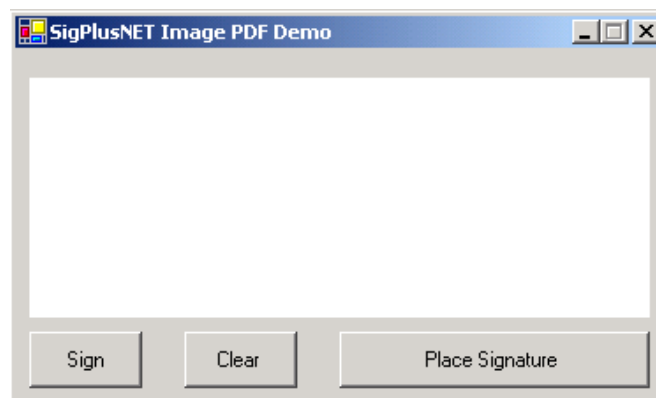
Topaz Systems, Inc. iTextSharp.NET PDF Signature Image How-To

Topaz Systems, Inc.
650 Cochran Street, Unit 6
Simi Valley, CA, 93065
©1995-2006, all rights reserved, US patent 6,307,955, and pending

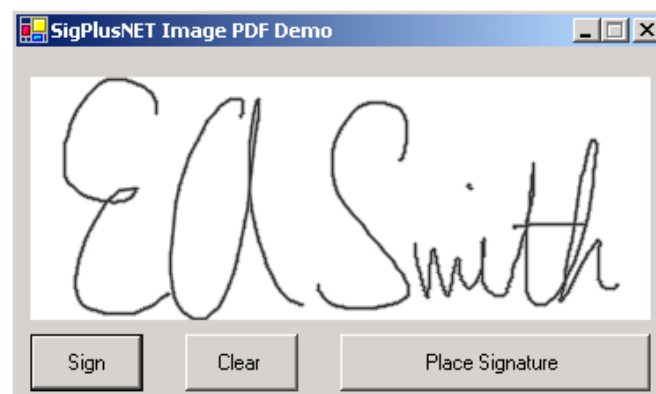
www.topazsystems.com
tech support: 805 520-8286
support@topazsystems.com

Welcome to the Topaz SigPlus.NET Signature Example Demo. This demonstration captures a signature and returns it to the developer as a buffered image. This image is then added to the PDF document. Because this demo will not create a biometric or a cryptographically-bound signature but instead creates a .jpeg image, the document is not legally binding.

Please run the C# demo found within the .NET package you downloaded from the website. The window below will appear.



Press the "Sign" button and sign the surface of your Topaz Signature Tablet. If you are unhappy with the signature, press "Clear", and resign. Once you like the signature, press "Place Signature" to place it into a PDF document. The document will not open, however it will be created and placed within the same folder as the application. The document will be saved as "SignatureTest.pdf", but this is customizable. Open "SignatureTest.pdf" to view the PDF document with signature attached.



Looking at the Code Behind the Demo

Next we will look at the C# code behind this demo. To access the code, open “SigPlusNET_iTextSharp.sln” which will be found in the same file folder as you found “SignatureTest.pdf”.

The code is very simple. The Sign button sets the tablet state to 1 to turn on the tablet and allow the computer to accept the signature data from the tablet. (SEE CODE BELOW)

```
private void cmdSign_Click(object sender, System.EventArgs e)
{
    SigPlusNET1.SetTabletState(1);
}
```

The code for the Clear button is equally simple. The ClearTablet() function call clears the ink out of the Topaz SigPlus object. (SEE CODE BELOW)

```
private void cmdClear_Click(object sender, System.EventArgs e)
{
    SigPlusNET1.ClearTablet();
}
```

The code behind the Place Signature Button is where the majority of the application takes place. (SEE BELOW AND NEXT PAGE))

```
private void cmdPlaceSignature_Click(object sender, System.EventArgs e)
{
    if (SigPlusNET1.NumberOfTabletPoints() == 0)
    {
        //User must sign first!!
    }
    else
    {
        SigPlusNET1.SetTabletState(0); //turn off signature pad
        //continue with PDF creation and add signature image
        // step 1: creation of a document-object
        Document document = new Document();

        try
        {
            // step 2:
            // we create a writer that listens to the document
            // and directs a PDF-stream to a file
            PdfWriter writer = PdfWriter.GetInstance(document, new FileStream("SignatureTest.pdf", FileMode.Create));

            // step 3: we open the document
            document.Open();

            // step 4: we add signature image to the document
            document.Add(new Phrase("Sample text for signature demo"));
        }
    }
}
```

```

System.Drawing.Image sigimage;

SigPlusNET1.SetImageXSize(500);
SigPlusNET1.SetImageYSize(150);
SigPlusNET1.SetJustifyY(10);
SigPlusNET1.SetJustifyX(10);
SigPlusNET1.SetJustifyMode(5);
SigPlusNET1.SetImagePenWidth(7);
SigPlusNET1.SetImageFileFormat(4); //0=bmp, 4=jpg, 6=tif
sigimage = SigPlusNET1.GetSigImage();

PdfContentByte cb = writer.DirectContent;
iTextSharp.text.Image image = iTextSharp.text.Image.GetInstance(sigimage, System.Drawing.Imaging.ImageFormat.Jpeg);

image.Transparency = new int[] {255, 255};
image.SetAbsolutePosition(100, 200);

cb.AddImage(image);

```

The “Place Signature” button turns the tablet state off and creates new Document and PDFWriter instances using the iTextSharp.DLL. It is in the instantiation of the PDFWriter that we specify the location of the PDF that we are to save. We open the PDF document and using SigPlus.NET size the signature image, justify it, and write it as a system.drawing.image in .jpeg format.

Next we use the iTextSharp assembly and write the image to the PDF. We set the transparency so that anything in the document behind the signature object will show through, and we can set the absolute position of the signature on the page. Finally the AddImage() function call adds the signature image to the PDF .

The iTextSharp.dll offers quite a bit more functionality in terms of PDF manipulations, should you want to do more work with your PDF. This demo simply shows how to add a signature .jpeg into the PDF programmatically, without writing the file first. For the latest information on the iTextSharp.dll please see <http://itextsharp.sourceforge.net/>.

For more information regarding products and support
Email: support@topazsystems.com
Phone: (805) 520-8286

www.topazsystems.com
650 Cochran Street, Unit 6, Simi Valley, CA, USA, 93065
Phone: 805 520-8282; Fax: 805 520-0867