www.bytescout.com

PDF SDK samples in ASP.NET using ByteScout PDF SDK

Make PDF SDK samples in ASP.NET

:

Tutorial on how to do PDF SDK samples in ASP.NET

ByteScout tutorials explain the material for programmers who use ASP.NET. ByteScout PDF SDK was made to help with PDF SDK samples in ASP.NET. ByteScout PDF SDK is the SDK for pdf documents generation, modification and updates. Can also generate and fill PDF forms. Provides support for text (fonts, style, size, font family), layers, pdf form fields, vector and raster drawings.

The SDK samples like this one below explain how to quickly make your application do PDF SDK samples in ASP.NET with the help of ByteScout PDF SDK. Follow the instruction from the scratch to work and copy and paste code for ASP.NET into your editor. Use of ByteScout PDF SDK in ASP.NET is also explained in the documentation included along with the product.

Trial version can be downloaded from our website. Source code samples for ASP.NET and documentation are included.

Source Code Files:

Default.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="PD

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>PDF SDK Samples</title>
</head>
<body>
    <form id="form1" runat="server">
        <div style="text-align:center;">
            <div style="font-size: 20px; text-decoration: underline;">
                <br /> This page demostrates Byscout PDF SDK, by perfomring following P
            </div>

            <div>
                <asp:Button ID="btnPDFConversionFromHtml" runat="server" Text="1. Html
                  <asp:Label ID="lblPDFConversationFromHTML" runat="server">
                <br />
                <br />
                <asp:Button ID="btnTableCreation" runat="server" Text="2. Table Creatio
                  <asp:Label ID="lblTableCreation" runat="server"></asp:Label
                <br />
                <br />
                <asp:Button ID="btnSplitPDF" runat="server" Text="3. Splitting PDF" Wid
                  <asp:Label ID="lblSplitPDF" runat="server"></asp:Label>
                <br />
                <br />
                <asp:Button ID="btnMergePDF" runat="server" Text="4. Merging PDF" Width
                  <asp:Label ID="lblMergePDF" runat="server"></asp:Label>
                <br />
                <br />
                <asp:Button ID="btnProtectingPDF" runat="server" Text="5. Protecting PD
                  <asp:Label ID="lblProtectingPDF" runat="server"></asp:Label
            </div>
        </div>
    </form>
</body>
</html>
```

Default.aspx.cs

```
using Bytescout.PDF;
using Bytescout.PDF.Converters;
```

```
using System;
using System.Drawing.Printing;
using System.IO;

namespace PDFSDKSamples
{
    public partial class Default : System.Web.UI.Page
    {

        /*
        IF YOU SEE TEMPORARY FOLDER ACCESS ERRORS:

        Temporary folder access is required for web application when you use ByteScout
        If you are getting errors related to the access to temporary folder like "Acces

        SOLUTION:

        If your IIS Application Pool has "Load User Profile" option enabled the IIS pr

        If you are running Web Application under an impersonated account or IIS_IUSRS

        In this case
        - check the User or User Group your web application is running under
        - then add permissions for this User or User Group to read and write into that
        - restart your web application and try again

        */

        #region Events

        /// <summary>
        /// Handle HTML to PDF conversation
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        protected void btnPDFConversionFromHtml_Click(object sender, EventArgs e)
        {
            try
            {
                // HTML to PDF Conversion
                using (HtmlToPdfConverter converter = new HtmlToPdfConverter())
                {
                    converter.PageSize = PaperKind.A4;
                    converter.Orientation = Bytescout.PDF.Converters.PaperOrientation.
                    converter.Footer = "<p style=\"color: blue;\">FOOTER TEXT</p>";

                    // Get html document in input stream
                    FileStream inputFileStream = new FileStream(Server.MapPath("~/Samp

                    // Define output stream
                    MemoryStream outputStream = new MemoryStream();

                    // Get converted PDF docuement in output stream
                    converter.ConvertHtmlToPdf(inputFileStream, outputStream);

                    // Download converted document
                    Response.Clear();
                    Response.ClearHeaders();

                    Response.AppendHeader("Content-Length", outputStream.Length.ToStri
```

```
                Response.ContentType = "text/pdf";
                Response.AppendHeader("Content-Disposition", "attachment;filename=

                Response.BinaryWrite(outputStream.ToArray());
                Response.End();
            }
        }
        catch (Exception ex)
        {
            lblPDFConversationFromHTML.Text = "Error: " + ex.Message;
        }
    }

    /// <summary>
    /// Handle Table Creation with PDF SDK
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    protected void btnTableCreation_Click(object sender, EventArgs e)
    {
        try
        {
            // Create new document
            Document pdfDocument = new Document();
            pdfDocument.RegistrationName = "demo";
            pdfDocument.RegistrationKey = "demo";

            // Add page
            Bytescout.PDF.Page page = new Bytescout.PDF.Page(PaperFormat.A4);
            pdfDocument.Pages.Add(page);

            DeviceColor lightGrayColor = new ColorGray(200);
            DeviceColor whiteColor = new ColorGray(255);
            DeviceColor lightBlueColor = new ColorRGB(200, 200, 250);
            DeviceColor lightRedColor = new ColorRGB(255, 200, 200);

            // Create a table and set default background color
            Bytescout.PDF.Table table = new Bytescout.PDF.Table();
            table.BackgroundColor = lightGrayColor;

            // Add row headers column and set its color
            table.Columns.Add(new TableColumn("RowHeaders"));
            table.Columns[0].BackgroundColor = lightGrayColor;

            // Add columns A, B, C, ...
            for (int c = 0; c < 10; c++)
            {
                string columnName = Convert.ToChar('A' + c).ToString();
                table.Columns.Add(new TableColumn(columnName, columnName));
            }

            // Add rows
            for (int r = 0; r < 10; r++)
            {
                // Create new row and set its background color
                Bytescout.PDF.TableRow row = table.NewRow();
                row.BackgroundColor = whiteColor;

                // Set row header text
                row["RowHeaders"].Text = (r + 1).ToString();
```

```
                // Set cell text
                for (int c = 0; c < 10; c++)
                {
                    string columnName = Convert.ToChar('A' + c).ToString();
                    row[columnName].Text = columnName + (r + 1);
                }

                // Add the row to the table
                table.Rows.Add(row);
            }

            // Decorate the table
            table.Rows[1]["B"].BackgroundColor = lightRedColor;
            table.Columns[2].BackgroundColor = lightBlueColor;
            table.Rows[1].BackgroundColor = lightBlueColor;
            table.Rows[1]["RowHeaders"].BackgroundColor = lightBlueColor;

            // Draw the table on canvas
            page.Canvas.DrawTable(table, 20, 20);

            // Save created PDF to memory stream
            MemoryStream memoryStream = new MemoryStream();
            pdfDocument.Save(memoryStream);

            // Perform download of file
            Response.Clear();
            Response.ClearHeaders();

            Response.AppendHeader("Content-Length", memoryStream.Length.ToString()
            Response.ContentType = "text/pdf";
            Response.AppendHeader("Content-Disposition", "attachment;filename=\"sa

            Response.BinaryWrite(memoryStream.ToArray());
            Response.End();
        }
        catch (Exception ex)
        {
            lblTableCreation.Text = "Error: " + ex.Message;
        }
    }

    /// <summary>
    /// Demonstrate Splitting PDF
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    protected void btnSplitPDF_Click(object sender, EventArgs e)
    {
        try
        {
            // Open Document
            Document document = new Document(Server.MapPath("~/SampleFiles/sample.p
            document.RegistrationName = "demo";
            document.RegistrationKey = "demo";

            // Create Split PDF Document
            Document documentSplitPDF = new Document();
            documentSplitPDF.RegistrationName = "demo";
            documentSplitPDF.RegistrationKey = "demo";
```

```
            // Get page 1&2 to Split PDF document
            for (int i = 0; i < 2; i++)
            {
                documentSplitPDF.Pages.Add(document.Pages[i]);
            }

            // Save splitted PDF to memory stream
            MemoryStream memoryStream = new MemoryStream();
            documentSplitPDF.Save(memoryStream);

            // Perform download of file
            Response.Clear();
            Response.ClearHeaders();

            Response.AppendHeader("Content-Length", memoryStream.Length.ToString()
            Response.ContentType = "text/pdf";
            Response.AppendHeader("Content-Disposition", "attachment;filename=\"sa

            Response.BinaryWrite(memoryStream.ToArray());
            Response.End();
        }
        catch (Exception ex)
        {
            lblSplitPDF.Text = "Error: " + ex.Message;
        }
    }

    /// <summary>
    /// Handle Merging PDF
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    protected void btnMergePDF_Click(object sender, EventArgs e)
    {
        try
        {
            // Open first document
            Document document1 = new Document(Server.MapPath("~/SampleFiles/sample
            document1.RegistrationName = "demo";
            document1.RegistrationKey = "demo";

            // Open second document
            Document document2 = new Document(Server.MapPath("~/SampleFiles/sample2
            document2.RegistrationName = "demo";
            document2.RegistrationKey = "demo";

            // Add pages from document2 to document1
            for (int i = 0; i < document2.Pages.Count; ++i)
            {
                document1.Pages.Add(document2.Pages[i]);
            }

            // Save merged PDF to memory stream
            MemoryStream memoryStream = new MemoryStream();
            document1.Save(memoryStream);

            // Perform download of file
            Response.Clear();
            Response.ClearHeaders();
```

```
            Response.AppendHeader("Content-Length", memoryStream.Length.ToString()
            Response.ContentType = "text/pdf";
            Response.AppendHeader("Content-Disposition", "attachment;filename=\"sa

            Response.BinaryWrite(memoryStream.ToArray());
            Response.End();
        }
        catch (Exception ex)
        {
            lblMergePDF.Text = "Error: " + ex.Message;
        }
    }

    /// <summary>
    /// Handle Protecting PDF
    /// </summary>
    /// <param name="sender"></param>
    /// <param name="e"></param>
    protected void btnProtectingPDF_Click(object sender, EventArgs e)
    {
        try
        {
            using (Document pdfDocument = new Document())
            {
                // Set registration key and password
                pdfDocument.RegistrationKey = "demo";
                pdfDocument.RegistrationName = "demo";

                // PDF file path
                string pdfFilePath = Server.MapPath("~/SampleFiles/sample.pdf");

                // Load pdf file
                pdfDocument.Load(pdfFilePath);

                // Set document encryption algorythm
                pdfDocument.Security.EncryptionAlgorithm = EncryptionAlgorithm.RC4_

                // Set various user permissions
                pdfDocument.Security.AllowPrintDocument = false;
                pdfDocument.Security.AllowContentExtraction = false;
                pdfDocument.Security.AllowModifyAnnotations = false;
                pdfDocument.Security.PrintQuality = PrintQuality.LowResolution;

                // PDF format supports two kinds of passwords: owner and user passw
                // User password allows to view document and perform allowed actio
                // Owner password allows everything, including changing passwords

                // Set owner password
                // pdfDocument.Security.OwnerPassword = "ownerpassword";

                // Set user password
                pdfDocument.Security.UserPassword = "password1";

                // Extract PDF document to Stream
                MemoryStream memoryStream = new MemoryStream();
                pdfDocument.Save(memoryStream);

                // Perform download of file
                Response.Clear();
```

```
                    Response.ClearHeaders();

                    Response.AppendHeader("Content-Length", memoryStream.Length.ToStri
                    Response.ContentType = "text/pdf";
                    Response.AppendHeader("Content-Disposition", "attachment;filename=

                    Response.BinaryWrite(memoryStream.ToArray());
                    Response.End();
                }
            }
            catch (Exception ex)
            {
                lblProtectingPDF.Text = "Error: " + ex.Message;
            }

        }

        #endregion
    }
}
```

Default.aspx.designer.cs

```
//------------------------------------------------------------------------------
// <auto-generated>
//     This code was generated by a tool.
//
//     Changes to this file may cause incorrect behavior and will be lost if
//     the code is regenerated.
// </auto-generated>
//------------------------------------------------------------------------------

namespace PDFSDKSamples {


    public partial class Default {

        /// <summary>
        /// form1 control.
        /// </summary>
        /// <remarks>
        /// Auto-generated field.
        /// To modify move field declaration from designer file to code-behind file.
        /// </remarks>
        protected global::System.Web.UI.HtmlControls.HtmlForm form1;

        /// <summary>
        /// btnPDFConversionFromHtml control.
        /// </summary>
        /// <remarks>
        /// Auto-generated field.
```

```
        /// To modify move field declaration from designer file to code-behind file.
        /// </remarks>
        protected global::System.Web.UI.WebControls.Button btnPDFConversionFromHtml;

        /// <summary>
        /// lblPDFConversationFromHTML control.
        /// </summary>
        /// <remarks>
        /// Auto-generated field.
        /// To modify move field declaration from designer file to code-behind file.
        /// </remarks>
        protected global::System.Web.UI.WebControls.Label lblPDFConversationFromHTML;

        /// <summary>
        /// btnTableCreation control.
        /// </summary>
        /// <remarks>
        /// Auto-generated field.
        /// To modify move field declaration from designer file to code-behind file.
        /// </remarks>
        protected global::System.Web.UI.WebControls.Button btnTableCreation;

        /// <summary>
        /// lblTableCreation control.
        /// </summary>
        /// <remarks>
        /// Auto-generated field.
        /// To modify move field declaration from designer file to code-behind file.
        /// </remarks>
        protected global::System.Web.UI.WebControls.Label lblTableCreation;

        /// <summary>
        /// btnSplitPDF control.
        /// </summary>
        /// <remarks>
        /// Auto-generated field.
        /// To modify move field declaration from designer file to code-behind file.
        /// </remarks>
        protected global::System.Web.UI.WebControls.Button btnSplitPDF;

        /// <summary>
        /// lblSplitPDF control.
        /// </summary>
        /// <remarks>
        /// Auto-generated field.
        /// To modify move field declaration from designer file to code-behind file.
        /// </remarks>
        protected global::System.Web.UI.WebControls.Label lblSplitPDF;

        /// <summary>
        /// btnMergePDF control.
        /// </summary>
        /// <remarks>
        /// Auto-generated field.
        /// To modify move field declaration from designer file to code-behind file.
        /// </remarks>
        protected global::System.Web.UI.WebControls.Button btnMergePDF;

        /// <summary>
        /// lblMergePDF control.
```

```
        /// </summary>
        /// <remarks>
        /// Auto-generated field.
        /// To modify move field declaration from designer file to code-behind file.
        /// </remarks>
        protected global::System.Web.UI.WebControls.Label lblMergePDF;

        /// <summary>
        /// btnProtectingPDF control.
        /// </summary>
        /// <remarks>
        /// Auto-generated field.
        /// To modify move field declaration from designer file to code-behind file.
        /// </remarks>
        protected global::System.Web.UI.WebControls.Button btnProtectingPDF;

        /// <summary>
        /// lblProtectingPDF control.
        /// </summary>
        /// <remarks>
        /// Auto-generated field.
        /// To modify move field declaration from designer file to code-behind file.
        /// </remarks>
        protected global::System.Web.UI.WebControls.Label lblProtectingPDF;
    }
}
```

PDFSDKSamples.sln

```
Microsoft Visual Studio Solution File, Format Version 12.00
# Visual Studio 15
VisualStudioVersion = 15.0.27703.2026
MinimumVisualStudioVersion = 10.0.40219.1
Project("{FAE04EC0-301F-11D3-BF4B-00C04F79EFBC}") = "PDFSDKSamples", "PDFSDKSamples.csp
EndProject
Global
        GlobalSection(SolutionConfigurationPlatforms) = preSolution
                Debug|Any CPU = Debug|Any CPU
                Release|Any CPU = Release|Any CPU
        EndGlobalSection
        GlobalSection(ProjectConfigurationPlatforms) = postSolution
                {762F5B5F-2A6B-46F8-9B22-6107C2514B41}.Debug|Any CPU.ActiveCfg = Debug
                {762F5B5F-2A6B-46F8-9B22-6107C2514B41}.Debug|Any CPU.Build.0 = Debug|An
                {762F5B5F-2A6B-46F8-9B22-6107C2514B41}.Release|Any CPU.ActiveCfg = Rele
                {762F5B5F-2A6B-46F8-9B22-6107C2514B41}.Release|Any CPU.Build.0 = Releas
        EndGlobalSection
        GlobalSection(SolutionProperties) = preSolution
                HideSolutionNode = FALSE
        EndGlobalSection
```

```
        GlobalSection(ExtensibilityGlobals) = postSolution
                SolutionGuid = {912E8124-8E79-49D2-BD0A-364AA7D726DA}
        EndGlobalSection
EndGlobal
```

Web.config

```
<?xml version="1.0"?>
<!--
  For more information on how to configure your ASP.NET application, please visit
  https://go.microsoft.com/fwlink/?LinkId=169433
  -->
<configuration>
  <!--
    For a description of web.config changes see http://go.microsoft.com/fwlink/?LinkId=

    The following attributes can be set on the <httpRuntime> tag.
      <system.Web>
        <httpRuntime targetFramework="4.5" />
      </system.Web>
  -->
  <system.web>
    <compilation debug="true" targetFramework="4.5"/>
    <pages controlRenderingCompatibilityVersion="4.0"/>
  </system.web>
</configuration>
```

VIDEO

https://www.youtube.com/watch?v=gdsQ0EAqwGQ

ON-PREMISE OFFLINE SDK