

How to split pdf by barcode with barcode reader sdk in ASP.NET C# and ByteScout Premium Suite

This code in ASP.NET C# shows how to split pdf by barcode with barcode reader sdk with this how to tutorial

The code displayed below will guide you to install an ASP.NET C# app to split pdf by barcode with barcode reader sdk. ByteScout Premium Suite is the set that includes 12 SDK products from ByteScout including tools and components for PDF, barcodes, spreadsheets, screen video recording and you can use it to split pdf by barcode with barcode reader sdk with ASP.NET C#.

The following code snippet for ByteScout Premium Suite works best when you need to quickly split pdf by barcode with barcode reader sdk in your ASP.NET C# application. IF you want to implement the functionality, just copy and paste this code for ASP.NET C# below into your code editor with your app, compile and run your application. Applying ASP.NET C# application mostly includes various stages of the software development so even if the functionality works please test it with your data and the production environment.

ByteScout provides the free trial version of ByteScout Premium Suite along with the documentation and source code samples.

FOR MORE INFORMATION AND FREE TRIAL:

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

[Read more about ByteScout Premium Suite](#)

[Explore API Documentation](#)

[Get Free Training for ByteScout Premium Suite](#)

[Get Free API key for Web API](#)

[visit www.ByteScout.com](http://www.ByteScout.com)

Source Code Files:

Default.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="WebBarcodeReaderTester.Default" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
<title>Web Barcode Reader Tester (C#)</title>
</head>
<body>
<form id="form1" runat="server">
<div>
Click browse button to upload a PDF document with a barcode inside so this PDF
Choose sample "barcodes.pdf" file:
<br />
<asp:Label ID="LabelInputFile" runat="server" Text="LabelInputFile"></asp:Label>
<asp:FileUpload ID="FileUpload1" runat="server" /><br />
<br />
<asp:Button id="UploadButton"
Text="Upload And Split File"
OnClick="UploadButton_Click"
runat="server">
</asp:Button>
<br />
<br />
<asp:Label id="UploadStatusLabel" Text="" runat="server"></asp:Label>
<br />
<asp:ListBox ID="ListBox1" runat="server" Visible="False"></asp:ListBox><br />
<br />
</div>
</form>
</body>
</html>
```

Default.aspx.cs

```
using System;
using System.IO;
using Bytescout.BarCodeReader;

namespace WebTestSharp
{
    /*
    IF YOU SEE TEMPORARY FOLDER ACCESS ERRORS:
```

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

SOLUTION:

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

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

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 temp
- restart your web application and try again

*/

```
public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        LabelInputFile.Text = MapPath("barcodes.pdf");
    }

    protected void UploadButton_Click(object sender, EventArgs e)
    {
        String savePath = @"\uploads\";

        if (FileUpload1.HasFile)
        {
            String fileName = FileUpload1.FileName;
            fileName = savePath + fileName;
            // remove if already exists
            if (File.Exists(fileName))
                File.Delete(fileName);

            FileUpload1.SaveAs(Server.MapPath(fileName));

            Reader barcodeReader = new Reader();

            // Limit search to 1D barcodes only (exclude 2D barcodes)
            // Change to barcodeReader.BarcodeTypesToFind.SetAll()
            // or select specific type, e.g. barcodeReader.BarcodeTypesToFind.SetAll1D();
            barcodeReader.BarcodeTypesToFind.SetAll1D();

            // reader.MediumTrustLevelCompatible = true; // uncomment
            UploadStatusLabel.Visible = false;

            ListBox1.Items.Clear();
            ListBox1.Visible = true;
            ListBox1.Items.Add("Searching for barcode(s) from PDF file");

            FoundBarcode[] barcodes = barcodeReader.ReadFrom(Server.MapPath(fileName));

            if (barcodes.Length == 0)
            {
                ListBox1.Items.Add("No barcodes found");
            }
            else
            {
                // saving filename of the pdf file
            }
        }
    }
}
```

```

string lastFilename = fileName;

int partIndex = 0;

                foreach (FoundBarcode barcode in barcodes)
                {
                    // Split PDF document in two parts by found barcode
                    partIndex++;
                    barcodeReader.SplitDocument(
                        Server.MapPath(lastFilename),
                        Server.MapPath(savePath+"part-" + partIndex + ".pdf"),
                        Server.MapPath(savePath + @"part-last.pdf"),
                        barcode.Page + 1);

                    ListBox1.Items.Add(String.Format("Splitting " + fileName + " f
                    ListBox1.Items.Add("Saved:" + savePath + @"part-" + partIndex -

                    lastFilename = savePath + @"part-last.pdf";

                }
            }
        }
    }
}

```

Default.aspx.designer.cs

```

//-----
// <auto-generated>
//     This code was generated by a tool.
//     Runtime Version:2.0.50727.8009
//
//     Changes to this file may cause incorrect behavior and will be lost if
//     the code is regenerated.
// </auto-generated>
//-----

namespace WebTestSharp {

    /// <summary>
    /// _Default class.
    /// </summary>
    /// <remarks>

```

```
/// Auto-generated class.
/// </remarks>
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>
    /// LabelInputFile 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 LabelInputFile;

    /// <summary>
    /// FileUpload1 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.FileUpload FileUpload1;

    /// <summary>
    /// UploadButton 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 UploadButton;

    /// <summary>
    /// UploadStatusLabel 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 UploadStatusLabel;

    /// <summary>
    /// ListBox1 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.ListBox ListBox1;
}
}
```

Web.config

```
<?xml version="1.0"?>
<configuration>
  <appSettings/>
  <connectionStrings/>
  <system.web>
    <!--
      Set compilation debug="true" to insert debugging
      symbols into the compiled page. Because this
      affects performance, set this value to true only
      during development.
    -->
    <compilation debug="true" />
    <!--
      The <authentication> section enables configuration
      of the security authentication mode used by
      ASP.NET to identify an incoming user.
    -->
    <authentication mode="Windows" />
    <!--
      The <customErrors> section enables configuration
      of what to do if/when an unhandled error occurs
      during the execution of a request. Specifically,
      it enables developers to configure html error pages
      to be displayed in place of a error stack trace.
    -->
    <customErrors mode="RemoteOnly" defaultRedirect="GenericErrorPage.htm">
      <error statusCode="403" redirect="NoAccess.htm" />
      <error statusCode="404" redirect="FileNotFound.htm" />
    </customErrors>
  </system.web>
</configuration>
```

VIDEO

<https://www.youtube.com/watch?v=NEwNs2b9YN8>

ON-PREMISE OFFLINE SDK

[60 Day Free Trial](#) or [Visit ByteScout Premium Suite Home Page](#)

[Explore ByteScout Premium Suite Documentation](#)

[Explore Samples](#)

[Sign Up for ByteScout Premium Suite Online Training](#)

ON-DEMAND REST WEB API

[Get Your API Key](#)

[Explore Web API Docs](#)

[Explore Web API Samples](#)

[visit www.ByteScout.com](http://www.ByteScout.com)

[visit www.PDF.co](http://www.PDF.co)

www.bytescout.com