

www.bytescout.com

How to decode page by page from pdf or tiff with barcode reader sdk in ASP.NET C# with ByteScout Data Extraction Suite

If you want to learn more then this tutorial will show how to decode page by page from pdf or tiff with barcode reader sdk in ASP.NET C#

Every ByteScout tool includes simple example ASP.NET C# source codes that you can get here or in the folder with installed ByteScout product. ByteScout Data Extraction Suite is the set that includes 3 SDK products for data extraction from PDF, scans, images and from spreadsheets: PDF Extractor SDK, Data Extraction SDK, Barcode Reader SDK. It can be applied to decode page by page from pdf or tiff with barcode reader sdk using ASP.NET C#.

The SDK samples given below describe how to quickly make your application do decode page by page from pdf or tiff with barcode reader sdk in ASP.NET C# with the help of ByteScout Data Extraction Suite. Just copy and paste the code into your ASP.NET C# application's code and follow the instructions. Use of ByteScout Data Extraction Suite in ASP.NET C# is also described in the documentation included along with the product.

You can download free trial version of ByteScout Data Extraction Suite from our website to see and try many others source code samples for ASP.NET C#.

FOR MORE INFORMATION AND FREE TRIAL:

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

[Read more about ByteScout Data Extraction Suite](#)

[Explore API Documentation](#)

[Get Free Training for ByteScout Data Extraction Suite](#)

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

[visit 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/TIFF document<br />
            <asp:FileUpload ID="FileUpload1" runat="server" /><br />
            <br />
            Page No. to read barcode from:<br />
            <asp:TextBox ID="txtPageNo" Text="1" runat="server"></asp:TextBox>
            <br />
            <br />
            <asp:Button ID="UploadButton" runat="server" Text="Upload file and read barcodes" OnClick="UploadButton_Click" />
            <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 Bytescout.BarCodeReader;

namespace WebTestSharp
{
    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 API. If you are getting errors related to the access to temporary folder like "Access is denied" or "The process cannot access the file because it is being used by another process", it means that your application does not have enough permissions to write to the temporary folder.

SOLUTION:

If your IIS Application Pool has "Load User Profile" option enabled the IIS process will run under the context of the user who is logged in.

If you are running Web Application under an impersonated account or IIS_IUSRS group, then you will need to grant permissions to that account.

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 folder
- restart your web application and try again

*/

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

    if (FileUpload1.HasFile)
    {
        String fileName = FileUpload1.FileName;
        savePath += fileName;
        FileUpload1.SaveAs(Server.MapPath(savePath));

        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();

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

        // Page No to read
        int pageNoToRead = Convert.ToInt32(txtPageNo.Text);

        ListBox1.Items.Clear();
        ListBox1.Visible = true;
        ListBox1.Items.Add("Reading barcode(s) from file \\" + savePath + "\\");

        // Reading barcode from document on page-by-page basic
        FoundBarcode[] barcodes = barcodeReader.ReadFrom(Server.MapPath(savePath), pageNoToRead);

        if (barcodes.Length == 0)
        {
            ListBox1.Items.Add("No barcodes found");
        }
        else
        {
            foreach (FoundBarcode barcode in barcodes)
            {
                ListBox1.Items.Add(String.Format("Found barcode - Type: '{0}'", barcode.Type));
            }
        }
    }
}
```

```
        // Notify the user that a file was not uploaded.
        UploadStatusLabel.Text = "You did not specify a file to upload." ;
    }
}
```

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 WebTestSharp {

    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>
        /// 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>
        /// txtPageNo 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.TextBox txtPageNo;
    }
}
```

```

    ///<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>
        <httpRuntime maxRequestLength="1048576" />

        <!--
            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="NotFound.htm" />
</customErrors>
-->
</system.web>
</configuration>
```

VIDEO

<https://www.youtube.com/watch?v=NEWNs2b9YN8>

ON-PREMISE OFFLINE SDK

[60 Day Free Trial](#) or [Visit ByteScout Data Extraction Suite Home Page](#)
[Explore ByteScout Data Extraction Suite Documentation](#)
[Explore Samples](#)
[Sign Up for ByteScout Data Extraction Suite Online Training](#)

ON-DEMAND REST WEB API

[Get Your API Key](#)
[Explore Web API Docs](#)
[Explore Web API Samples](#)

[visit www.ByteScout.com](#)

[visit www.PDF.co](#)

www.bytescout.com