www.bytescout.com

How to convert PDF to XLS from uploaded file (winforms) for PDF to excel API in C# with ByteScout Cloud API Server

Step By Step Instructions on how to convert PDF to XLS from uploaded file (winforms) for PDF to excel API in C#

This page displays the step-by-step instructions and algorithm of how to convert PDF to XLS from uploaded file (winforms) and how to apply it in your application. ByteScout Cloud API Server was designed to assist PDF to excel API in C#. ByteScout Cloud API Server is the ready to deploy Web API Server that can be deployed in less than thirty minutes into your own in-house Windows server (no Internet connnection is required to process data!) or into private cloud server. Can store data on in-house local server based storage or in Amazon AWS S3 bucket. Processing data solely on the server using built-in ByteScout powered engine, no cloud services are used to process your data!.

If you want to speed up the application's code writing then C# code samples for C# developers help to implement using ByteScout Cloud API Server. Open your C# project and simply copy & paste the code and then run your app! You can use these C# sample examples in one or many applications.

Our website provides free trial version of ByteScout Cloud API Server that gives source code samples to assist with your C# project.

Source Code Files:

Form1.Designer.cs

```
namespace PdfToExcelFrom
{
    partial class Form1
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; other
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.label1 = new System.Windows.Forms.Label();
            this.label2 = new System.Windows.Forms.Label();
            this.openFileDialog1 = new System.Windows.Forms.OpenFileDialog();
            this.txtInputPDFFileName = new System.Windows.Forms.TextBox();
            this.btnSelectInputFile = new System.Windows.Forms.Button();
            this.label3 = new System.Windows.Forms.Label();
            this.cmbOutputAs = new System.Windows.Forms.ComboBox();
            this.btnConvert = new System.Windows.Forms.Button();
            this.label4 = new System.Windows.Forms.Label();
            this.cmbConvertTo = new System.Windows.Forms.ComboBox();
            this.SuspendLayout();
            //
            // label2
            //
            this.label2.AutoSize = true;
            this.label2.Location = new System.Drawing.Point(13, 51);
            this.label2.Name = "label2";
            this.label2.Size = new System.Drawing.Size(96, 17);
            this.label2.TabIndex = 2;
            this.label2.Text = "Input PDF File";
            //
            // openFileDialog1
            //
```

```
            this.openFileDialog1.FileName = "openFileDialog1";
            this.openFileDialog1.FileOk += new System.ComponentModel.CancelEventHandle
            //
            // txtInputPDFFileName
            //
            this.txtInputPDFFileName.Location = new System.Drawing.Point(115, 48);
            this.txtInputPDFFileName.Name = "txtInputPDFFileName";
            this.txtInputPDFFileName.Size = new System.Drawing.Size(290, 22);
            this.txtInputPDFFileName.TabIndex = 3;
            //
            // btnSelectInputFile
            //
            this.btnSelectInputFile.Font = new System.Drawing.Font("Microsoft Sans Ser
            this.btnSelectInputFile.Location = new System.Drawing.Point(411, 46);
            this.btnSelectInputFile.Name = "btnSelectInputFile";
            this.btnSelectInputFile.Size = new System.Drawing.Size(126, 36);
            this.btnSelectInputFile.TabIndex = 4;
            this.btnSelectInputFile.Text = "Select input File";
            this.btnSelectInputFile.UseVisualStyleBackColor = true;
            this.btnSelectInputFile.Click += new System.EventHandler(this.btnSelectInpu
            //
            // label3
            //
            this.label3.AutoSize = true;
            this.label3.Location = new System.Drawing.Point(13, 133);
            this.label3.Name = "label3";
            this.label3.Size = new System.Drawing.Size(71, 17);
            this.label3.TabIndex = 5;
            this.label3.Text = "Output As";
            //
            // cmbOutputAs
            //
            this.cmbOutputAs.FormattingEnabled = true;
            this.cmbOutputAs.Items.AddRange(new object[] {
            "URL to output file",
            "inline content"});
            this.cmbOutputAs.Location = new System.Drawing.Point(107, 133);
            this.cmbOutputAs.Name = "cmbOutputAs";
            this.cmbOutputAs.Size = new System.Drawing.Size(430, 24);
            this.cmbOutputAs.TabIndex = 6;
            //
            // btnConvert
            //
            this.btnConvert.Location = new System.Drawing.Point(16, 176);
            this.btnConvert.Name = "btnConvert";
            this.btnConvert.Size = new System.Drawing.Size(194, 43);
            this.btnConvert.TabIndex = 7;
            this.btnConvert.Text = "Convert";
            this.btnConvert.UseVisualStyleBackColor = true;
            this.btnConvert.Click += new System.EventHandler(this.btnConvert_Click);
            //
            // label4
            //
            this.label4.AutoSize = true;
            this.label4.Location = new System.Drawing.Point(13, 93);
            this.label4.Name = "label4";
            this.label4.Size = new System.Drawing.Size(78, 17);
            this.label4.TabIndex = 8;
            this.label4.Text = "Convert To";
            //
```

```
            // cmbConvertTo
            //
            this.cmbConvertTo.FormattingEnabled = true;
            this.cmbConvertTo.Items.AddRange(new object[] {
            "XLS",
            "XLSX"});
            this.cmbConvertTo.Location = new System.Drawing.Point(107, 93);
            this.cmbConvertTo.Name = "cmbConvertTo";
            this.cmbConvertTo.Size = new System.Drawing.Size(430, 24);
            this.cmbConvertTo.TabIndex = 9;
            //
            // Form1
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(8F, 16F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(554, 255);
            this.Controls.Add(this.cmbConvertTo);
            this.Controls.Add(this.label4);
            this.Controls.Add(this.btnConvert);
            this.Controls.Add(this.cmbOutputAs);
            this.Controls.Add(this.label3);
            this.Controls.Add(this.btnSelectInputFile);
            this.Controls.Add(this.txtInputPDFFileName);
            this.Controls.Add(this.label2);
            this.Name = "Form1";
            this.Text = "Cloud API: PDF to Excel Conversion";
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.OpenFileDialog openFileDialog1;
        private System.Windows.Forms.TextBox txtInputPDFFileName;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.ComboBox cmbOutputAs;
        private System.Windows.Forms.Button btnConvert;
        private System.Windows.Forms.Button btnSelectInputFile;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.ComboBox cmbConvertTo;
    }
}
```

Form1.cs

```
using Newtonsoft.Json.Linq;
using System;
```

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Drawing;
using System.IO;
using System.Net;
using System.Text;
using System.Windows.Forms;

namespace PdfToExcelFrom
{
    // Please NOTE: In this sample we're assuming Cloud Api Server is hosted at "https
    // If it's not then please replace this with with your hosting url.
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        #region File Selection

        private void openFileDialog1_FileOk(object sender, CancelEventArgs e)
        {
            txtInputPDFFileName.Text = openFileDialog1.FileName;
        }

        private void btnSelectInputFile_Click(object sender, EventArgs e)
        {
            openFileDialog1.ShowDialog();
        }

        #endregion

        #region Convert PDF to Excel

        /// <summary>
        /// Perform convert PDF to Excel
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        private void btnConvert_Click(object sender, EventArgs e)
        {
            try
            {
                if (ValidateInputs())
                {
                    // Comma-separated list of page indices (or ranges) to process. Le
                    const string Pages = "";

                    // PDF document password. Leave empty for unprotected documents.
                    const string Password = "";

                    // Checks whether convert to as inline content.
                    bool isInline = Convert.ToString(cmbOutputAs.SelectedItem).ToLower

                    // Destination file name
                    string DestinationFile = string.Format(@".\{0}.{1}",
                        Path.GetFileNameWithoutExtension(txtInputPDFFileName.Text),
```

```
                Convert.ToString(cmbConvertTo.SelectedItem).ToLower());

            // Create standard .NET web client instance
            WebClient webClient = new WebClient();

            // 1. RETRIEVE THE PRESIGNED URL TO UPLOAD THE FILE.
            // * If you already have a direct file URL, skip to the step 3.

            // Prepare URL for `Get Presigned URL` API call
            string query = Uri.EscapeUriString(string.Format(
                "https://localhost/file/upload/get-presigned-url?contenttype=ap
                Path.GetFileName(txtInputPDFFileName.Text)));

            // Execute request
            string response = webClient.DownloadString(query);

            // Parse JSON response
            JObject json = JObject.Parse(response);

            if (json["error"].ToObject<bool>() == false)
            {
                // Get URL to use for the file upload
                string uploadUrl = json["presignedUrl"].ToString();
                string uploadedFileUrl = json["url"].ToString();

                // 2. UPLOAD THE FILE TO CLOUD.

                webClient.Headers.Add("content-type", "application/octet-strea
                webClient.UploadFile(uploadUrl, "PUT", txtInputPDFFileName.Tex
                webClient.Headers.Remove("content-type");

                // 3. CONVERT UPLOADED PDF FILE TO Excel

                // Prepare URL for `PDF To Excel` API call
                query = Uri.EscapeUriString(string.Format(
                    "https://localhost/pdf/convert/to/{4}?name={0}&password={1
                    Path.GetFileName(DestinationFile),
                    Password,
                    Pages,
                    uploadedFileUrl,
                    Convert.ToString(cmbConvertTo.SelectedItem).ToLower(),
                    isInline));

                // Execute request
                response = webClient.DownloadString(query);

                // Parse JSON response
                json = JObject.Parse(response);

                if (json["error"].ToObject<bool>() == false)
                {
                    // Get URL of generated Excel file
                    string resultFileUrl = json["url"].ToString();

                    // Download Excel output file
                    webClient.DownloadFile(resultFileUrl, DestinationFile);

                    MessageBox.Show({code}quot;Generated XLS file saved as {Des

                    // Open Downloaded output file
```

```
                            Process.Start(DestinationFile);
                        }
                        else
                        {
                            MessageBox.Show(json["message"].ToString());
                        }
                    }
                    else
                    {
                        MessageBox.Show(json["message"].ToString());
                    }
                    webClient.Dispose();
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Error");
            }

        }

        /// <summary>
        /// Validates form inputs
        /// </summary>
        /// <returns></returns>
        private bool ValidateInputs()
        {
            if (string.IsNullOrEmpty(txtInputPDFFileName.Text))
            {
                throw new Exception("Input PDf file must be selected/entered.");
            }

            if (!System.IO.File.Exists(txtInputPDFFileName.Text))
            {
                throw new Exception("Input file does not exits.");
            }

            if (System.IO.Path.GetExtension(txtInputPDFFileName.Text).ToLower() != ".pd
            {
                throw new Exception("Input file must be PDF");
            }

            if (string.IsNullOrEmpty(Convert.ToString(cmbConvertTo.SelectedItem)))
            {
                throw new Exception("Please select convert to option.");
            }

            if (string.IsNullOrEmpty(Convert.ToString(cmbOutputAs.SelectedItem)))
            {
                throw new Exception("Please select output-as option");
            }

            return true;
        }

        #endregion

    }
}
```

Program.cs

```
using System;
using System.Collections.Generic;
using System.Windows.Forms;

namespace PdfToExcelFrom
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

packages.config

```
<?xml version="1.0" encoding="utf-8"?>
<packages>
  <package id="Newtonsoft.Json" version="11.0.2" targetFramework="net20" />
</packages>
```

VIDEO

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

ON-PREMISE OFFLINE SDK

60 Day Free Trial or Visit ByteScout Cloud API Server Home Page
Explore ByteScout Cloud API Server Documentation
Explore Samples
Sign Up for ByteScout Cloud API Server 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