scanned PDF to JSON in C# using ByteScout PDF Extractor SDK

Make scanned PDF to JSON in C#

:

Tutorial on how to do scanned PDF to JSON in C#

Today we will explain the steps and algorithm of implementing scanned PDF to JSON and how to make it work in your application. Scanned PDF to JSON in C# can be implemented with ByteScout PDF Extractor SDK. ByteScout PDF Extractor SDK is the SDK is designed to help developers with pdf tables and pdf data extraction from unstructured documents like pdf, tiff, scans, images, scanned and electronic forms. The library is powered by OCR, computer vision and AI to provide unique functionality like table detection, automatic table structure extraction, data restoration, data restructuring and reconstruction. Supports PDF, TIFF, PNG, JPG images as input and can output CSV, XML, JSON formatted data. Includes full set of utilities like pdf splitter, pdf merger, searchable pdf maker and other utilities.

C# code snippet like this for ByteScout PDF Extractor SDK works best when you need to quickly implement scanned PDF to JSON in your C# application. In order to implement this functionality, you should copy and paste code below into your app using code editor. Then compile and run your application. Test C# sample code examples whether they respond your needs and requirements for the project.

Trial version can be obtained from our website for free. It includes this and other source code samples for C#.

Source Code Files:

Program.cs

```
using System.Diagnostics;
using Bytescout.PDFExtractor;

// This example demonstrates the use of Optical Character Recognition (OCR) to extract
// from scanned PDF documents and raster images.

// To make OCR work you should add the following references to your project:
// 'Bytescout.PDFExtractor.dll', 'Bytescout.PDFExtractor.OCRExtension.dll'.

namespace ScannedPdfToJSON
{
    class Program
    {
        static void Main(string[] args)
        {
            // Create Bytescout.PDFExtractor.JSONExtractor instance
            JSONExtractor extractor = new JSONExtractor();
            extractor.RegistrationName = "demo";
            extractor.RegistrationKey = "demo";

            // Load sample PDF document
            extractor.LoadDocumentFromFile("sample_ocr.pdf");

            // Enable Optical Character Recognition (OCR)
            // in .Auto mode (SDK automatically checks if needs to use OCR or not)
            extractor.OCRMode = OCRMode.Auto;

            // Set the location of OCR language data files
            extractor.OCRLanguageDataFolder = @"c:\Program Files\Bytescout PDF Extracto

            // Set OCR language
            extractor.OCRLanguage = "eng"; // "eng" for english, "deu" for German, "fre
            // Find more language files at https://github.com/bytescout/ocrdata

            // Set PDF document rendering resolution
            extractor.OCRResolution = 300;


            // You can also apply various preprocessing filters
            // to improve the recognition on low-quality scans.

            // Automatically deskew skewed scans
            //extractor.OCRImagePreprocessingFilters.AddDeskew();

            // Remove vertical or horizontal lines (sometimes helps to avoid OCR engine
            //extractor.OCRImagePreprocessingFilters.AddVerticalLinesRemover();
            //extractor.OCRImagePreprocessingFilters.AddHorizontalLinesRemover();

            // Repair broken letters
            //extractor.OCRImagePreprocessingFilters.AddDilate();
            // Remove noise
            //extractor.OCRImagePreprocessingFilters.AddMedian();

            // Apply Gamma Correction
            //extractor.OCRImagePreprocessingFilters.AddGammaCorrection();

            // Add Contrast
            //extractor.OCRImagePreprocessingFilters.AddContrast(20);


            // (!) You can use new OCRAnalyser class to find an optimal set of image pr
            // filters for your specific document.
            // See "OCR Analyser" example.


            // Save extracted text to file
            extractor.SaveJSONToFile("output.json");

            // Cleanup
            extractor.Dispose();

            // Open result document in default associated application (for demo purpose
            ProcessStartInfo processStartInfo = new ProcessStartInfo("output.json");
            processStartInfo.UseShellExecute = true;
            Process.Start(processStartInfo);
        }
    }
}
```

VIDEO

https://www.youtube.com/watch?v=s28W3_KMraU

ON-PREMISE OFFLINE SDK

60 Day Free Trial or Visit ByteScout PDF Extractor SDK Home Page
Explore ByteScout PDF Extractor SDK Documentation
Explore Samples
Sign Up for ByteScout PDF Extractor SDK Online Training

ON-DEMAND REST WEB API

Get Your API Key