How to convert PDF to JSON from uploaded file asynchronously for PDF to JSON API in C# and PDF.co Web API

Tutorial: how to convert PDF to JSON from uploaded file asynchronously for PDF to JSON API in C#

Today we will explain the steps and algorithm of how to convert PDF to JSON from uploaded file asynchronously and how to make it work in your application. PDF.co Web API was made to help with PDF to JSON API in C#. PDF.co Web API is the flexible Web API that includes full set of functions from e-signature requests to data extraction, OCR, images recognition, pdf splitting and pdf splitting. Can also generate barcodes and read barcodes from images, scans and pdf.

Fast application programming interfaces of PDF.co Web API for C# plus the instruction and the code below will help to learn how to convert PDF to JSON from uploaded file asynchronously. Sample code in C# is all you need. Copy-paste it to your the code editor, then add a reference to PDF.co Web API and you are ready to try it! Code testing will allow the function to be tested and work properly with your data.

Our website provides free trial version of PDF.co Web API that includes source code samples to help with your C# project.

C# - Program.cs

```
using System;
using System.IO;
using System.Net;
using Newtonsoft.Json.Linq;
using System.Threading;


// Cloud API asynchronous "PDF To JSON" job example.
// Allows to avoid timeout errors when processing huge or scanned PDF documents.

namespace ByteScoutWebApiExample
{
        class Program
        {
                // The authentication key (API Key).
                // Get your own by registering at
https://app.pdf.co/documentation/api
                const String API_KEY = "**********************************";

        // Source PDF file
        const string SourceFile = @".\sample.pdf";
        // Comma-separated list of page indices (or ranges) to process. Leave empty
for all pages. Example: '0,2-5,7-'.
```

```
        const string Pages = "";
                // PDF document password. Leave empty for unprotected documents.
                const string Password = "";
                // Destination JSON file name
                const string DestinationFile = @".\result.json";
                // (!) Make asynchronous job
                const bool Async = true;


                static void Main(string[] args)
                {
                        // Create standard .NET web client instance
                        WebClient webClient = new WebClient();

                        // Set API Key
                        webClient.Headers.Add("x-api-key", API_KEY);


            // Upload file
            string uploadedFileUrl = UploadFile(webClient, SourceFile);


            // Prepare URL for `PDF To JSON` API call
            string query = Uri.EscapeUriString(string.Format(
                                "https://api.pdf.co/v1/pdf/convert/to/json?name=
{0}&password={1}&pages={2}&url={3}&async={4}",
                                Path.GetFileName(DestinationFile),
                                Password,
                                Pages,
                uploadedFileUrl,
                                Async));

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

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

                                if (json["error"].ToObject() == false)
                                {
                                        // Asynchronous job ID
                                        string jobId = json["jobId"].ToString();
                                        // URL of generated JSON file that will
available after the job completion
                                        string resultFileUrl =
json["url"].ToString();

                                        // Check the job status in a loop.
                                        // If you don't want to pause the main thread
you can rework the code
                                        // to use a separate thread for the status
checking and completion.
                                        do
                                        {
                                                string status =
CheckJobStatus(webClient, jobId); // Possible statuses: "working", "failed",
"aborted", "success".
```

```
                                        // Display timestamp and status (for
demo purposes)

Console.WriteLine(DateTime.Now.ToLongTimeString() + ": " + status);

                                        if (status == "success")
                                        {
                                                // Download JSON file

webClient.DownloadFile(resultFileUrl, DestinationFile);

                                                Console.WriteLine("Generated
JSON file saved as \"{0}\" file.", DestinationFile);
                                                break;
                                        }
                                        else if (status == "working")
                                        {
                                                // Pause for a few seconds
                                                Thread.Sleep(3000);
                                        }
                                        else
                                        {
                                                Console.WriteLine(status);
                                                break;
                                        }
                                }
                                while (true);
                        }
                        else
                        {

Console.WriteLine(json["message"].ToString());
                        }
                }
                catch (WebException e)
                {
                        Console.WriteLine(e.ToString());
                }

                webClient.Dispose();


                Console.WriteLine();
                Console.WriteLine("Press any key...");
                Console.ReadKey();
        }

        static string UploadFile(WebClient webClient, string file)
        {
            // RETRIEVE THE PRESIGNED URL TO UPLOAD THE FILE:

            // Prepare URL for `Get Presigned URL` API call
            string query = Uri.EscapeUriString(string.Format(
                "https://api.pdf.co/v1/file/upload/get-presigned-url?
contenttype=application/octet-stream&name={0}",
                Path.GetFileName(SourceFile)));

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

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

            string uploadUrl = json["presignedUrl"].ToString();
            string uploadedFileUrl = json["url"].ToString();

            // UPLOAD FILE:

            webClient.Headers.Add("content-type", "application/octet-stream");
            webClient.UploadFile(uploadUrl, "PUT", SourceFile); // You can use
UploadData() instead if your file is byte[] or Stream
            webClient.Headers.Remove("content-type");

            return uploadedFileUrl;
        }

        static string CheckJobStatus(WebClient webClient, string jobId)
                {
                        string url = "https://api.pdf.co/v1/job/check?jobid=" +
jobId;

                        string response = webClient.DownloadString(url);
                        JObject json = JObject.Parse(response);

                    return Convert.ToString(json["status"]);
                }
        }
}
```

C# - packages.config

```
```

FOR MORE INFORMATION AND FREE TRIAL:

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

[Read more about PDF.co Web API]

Explore documentation

Visit www.ByteScout.com

or

Get Your Free API Key for www.PDF.co Web API