www.bytescout.com

PDF to image API in C# using PDF.co Web API

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.

FOR MORE INFORMATION AND FREE TRIAL:

Download Free Trial SDK (on-premise version)

Read more about PDF.co Web API

Explore API Documentation

Get Free Training for PDF.co Web API

Get Free API key for Web API

visit www.ByteScout.com

Source Code Files:

ByteScoutWebApiExample.csproj

```
<?xml version="1.0" encoding="utf-8"?>
<Project ToolsVersion="12.0" DefaultTargets="Build" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
  <Import Project="$(MSBuildExtensionsPath)\$(MSBuildToolsVersion)\Microsoft.Common.props" Condition="Exists('$
  <PropertyGroup>
    <Configuration Condition=" '$(Configuration)' == '' ">Debug</Configuration>
    <Platform Condition=" '$(Platform)' == '' ">AnyCPU</Platform>
    <ProjectGuid>{1E1C2C34-017E-4605-AE2B-55EA3313BE51}</ProjectGuid>
    <OutputType>Exe</OutputType>
    <RootNamespace>ByteScoutWebApiExample</RootNamespace>
    <AssemblyName>ByteScoutWebApiExample</AssemblyName>
    <TargetFrameworkVersion>v4.0</TargetFrameworkVersion>
    <FileAlignment>512</FileAlignment>
  </PropertyGroup>
  <PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Debug|AnyCPU' ">
    <PlatformTarget>AnyCPU</PlatformTarget>
    <DebugSymbols>true</DebugSymbols>
    <DebugType>full</DebugType>
```

```
    <Optimize>false</Optimize>
    <OutputPath>bin\Debug\</OutputPath>
    <DefineConstants>DEBUG;TRACE</DefineConstants>
    <ErrorReport>prompt</ErrorReport>
    <WarningLevel>4</WarningLevel>
  </PropertyGroup>
  <PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Release|AnyCPU' ">
    <PlatformTarget>AnyCPU</PlatformTarget>
    <DebugType>pdbonly</DebugType>
    <Optimize>true</Optimize>
    <OutputPath>bin\Release\</OutputPath>
    <DefineConstants>TRACE</DefineConstants>
    <ErrorReport>prompt</ErrorReport>
    <WarningLevel>4</WarningLevel>
  </PropertyGroup>
  <ItemGroup>
    <Reference Include="Newtonsoft.Json, Version=10.0.0.0, Culture=neutral, PublicKeyToken=30ad4fe6b2a6aeed, pr
      <HintPath>packages\Newtonsoft.Json.10.0.3\lib\net40\Newtonsoft.Json.dll</HintPath>
      <Private>True</Private>
    </Reference>
    <Reference Include="System" />
    <Reference Include="System.Core" />
    <Reference Include="System.Xml.Linq" />
    <Reference Include="System.Data" />
    <Reference Include="System.Xml" />
  </ItemGroup>
  <ItemGroup>
    <Compile Include="Program.cs" />
  </ItemGroup>
  <ItemGroup>
    <None Include="packages.config" />
    <None Include="sample.pdf">
      <CopyToOutputDirectory>Always</CopyToOutputDirectory>
    </None>
  </ItemGroup>
  <Import Project="$(MSBuildToolsPath)\Microsoft.CSharp.targets" />
  <!-- To modify your build process, add your task inside one of the targets below and uncomment it.
       Other similar extension points exist, see Microsoft.Common.targets.
  <Target Name="BeforeBuild">
  </Target>
  <Target Name="AfterBuild">
  </Target>
  -->
</Project>
```

ByteScoutWebApiExample.sln

```
Microsoft Visual Studio Solution File, Format Version 12.00
# Visual Studio 2013
VisualStudioVersion = 12.0.40629.0
MinimumVisualStudioVersion = 10.0.40219.1
Project("{FAE04EC0-301F-11D3-BF4B-00C04F79EFBC}") = "ByteScoutWebApiExample", "ByteScoutWebApiExamp
EndProject
Global
	GlobalSection(SolutionConfigurationPlatforms) = preSolution
		Debug|Any CPU = Debug|Any CPU
		Release|Any CPU = Release|Any CPU
	EndGlobalSection
	GlobalSection(ProjectConfigurationPlatforms) = postSolution
		{1E1C2C34-017E-4605-AE2B-55EA3313BE51}.Debug|Any CPU.ActiveCfg = Debug|Any CPU
		{1E1C2C34-017E-4605-AE2B-55EA3313BE51}.Debug|Any CPU.Build.0 = Debug|Any CPU
		{1E1C2C34-017E-4605-AE2B-55EA3313BE51}.Release|Any CPU.ActiveCfg = Release|Any CPU
		{1E1C2C34-017E-4605-AE2B-55EA3313BE51}.Release|Any CPU.Build.0 = Release|Any CPU
	EndGlobalSection
	GlobalSection(SolutionProperties) = preSolution
		HideSolutionNode = FALSE
```

```
        EndGlobalSection
EndGlobal
```

Program.cs

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


// Cloud API asynchronous "PDF To TIFF" 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 TIFF file name
        const string DestinationFile = @".\result.tif";
        // (!) Make asynchronous job
        const bool Async = true;
        // Advanced options
        const string Profiles = "{ 'profiles': [ { 'profile1': { 'TIFFCompression': 'CCITT4' } } ] }";

        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);

            // 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://api.pdf.co/v1/file/upload/get-presigned-url?contenttype=application/octet-stream&name={0}",
                Path.GetFileName(SourceFile)));

            try
            {
                // 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();
                    // Get URL of uploaded file to use with later API calls
                    string uploadedFileUrl = json["url"].ToString();
```

```
        // 2. UPLOAD THE FILE TO CLOUD.

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

        // 3. CONVERT UPLOADED PDF FILE TO TIFF

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

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

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

        if (json["error"].ToObject<bool>() == false)
        {
            // Asynchronous job ID
            string jobId = json["jobId"].ToString();
            // URL of generated TIFF 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(jobId); // Possible statuses: "working", "failed", "aborted", "success".

                // Display timestamp and status (for demo purposes)
                Console.WriteLine(DateTime.Now.ToLongTimeString() + ": " + status);

                if (status == "success")
                {
                    // Download TIFF file
                    webClient.DownloadFile(resultFileUrl, DestinationFile);

                    Console.WriteLine("Generated TIFF 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 CheckJobStatus(string jobId)
        {
            using (WebClient webClient = new WebClient())
            {
                // Set API Key
                webClient.Headers.Add("x-api-key", API_KEY);

                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"]);
            }
        }
    }
}
```

packages.config

```
<?xml version="1.0" encoding="utf-8"?>
<packages>
  <package id="Newtonsoft.Json" version="10.0.3" targetFramework="net40" />
</packages>
```

VIDEO

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

Explore Web API Samples

visit www.ByteScout.com

visit www.PDF.co