www.bytescout.com

PDF security in C# using PDF.co Web API

PDF.co Web API: the Web API with a set of tools for documents manipulation, data conversion, data extraction, splitting and merging of documents. Includes image recognition, built-in OCR, barcode generation and barcode decoders to decode bar codes from scans, pictures 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>
    <AppDesignerFolder>Properties</AppDesignerFolder>
    <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" />
    <Compile Include="Properties\AssemblyInfo.cs" />
  </ItemGroup>
  <ItemGroup>
    <None Include="packages.config" />
    <Content Include="sample.pdf">
      <CopyToOutputDirectory>Always</CopyToOutputDirectory>
    </Content>
  </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.CodeDom;
using System.Collections.Generic;
using System.IO;
using System.Net;
using Newtonsoft.Json;
using Newtonsoft.Json.Linq;

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";

        // Destination PDF file name
        const string DestinationFile = @".\protected.pdf";

        // Passwords to protect PDF document
        // The owner password will be required for document modification.
        // The user password only allows to view and print the document.
        const string OwnerPassword = "123456";
        const string UserPassword = "654321";

        // Encryption algorithm.
        // Valid values: "RC4_40bit", "RC4_128bit", "AES_128bit", "AES_256bit".
        const string EncryptionAlgorithm = "AES_128bit";

        // Allow or prohibit content extraction for accessibility needs.
        const bool AllowAccessibilitySupport = true;

        // Allow or prohibit assembling the document.
        const bool AllowAssemblyDocument = true;

        // Allow or prohibit printing PDF document.
        const bool AllowPrintDocument = true;

        // Allow or prohibit filling of interactive form fields (including signature fields) in PDF document.
        const bool AllowFillForms = true;

        // Allow or prohibit modification of PDF document.
        const bool AllowModifyDocument = true;

        // Allow or prohibit copying content from PDF document.
        const bool AllowContentExtraction = true;

        // Allow or prohibit interacting with text annotations and forms in PDF document.
        const bool AllowModifyAnnotations = true;

        // Allowed printing quality.
        // Valid values: "HighResolution", "LowResolution"
        const string PrintQuality = "HighResolution";
```

```
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 to the cloud
    string uploadedFileUrl = UploadFile(SourceFile);


    // PROTECT UPLOADED PDF DOCUMENT

    // Prepare requests params as JSON
    // See documentation: https://apidocs.pdf.co/?#pdf-security
    Dictionary<string, string> parameters = new Dictionary<string, string>();
    parameters.Add("name", Path.GetFileName(DestinationFile));
    parameters.Add("url", uploadedFileUrl);
    parameters.Add("ownerPassword", OwnerPassword);
    parameters.Add("userPassword", UserPassword);
    parameters.Add("encryptionAlgorithm", EncryptionAlgorithm);
    parameters.Add("allowAccessibilitySupport", AllowAccessibilitySupport.ToString());
    parameters.Add("allowAssemblyDocument", AllowAssemblyDocument.ToString());
    parameters.Add("allowPrintDocument", AllowPrintDocument.ToString());
    parameters.Add("allowFillForms", AllowFillForms.ToString());
    parameters.Add("allowModifyDocument", AllowModifyDocument.ToString());
    parameters.Add("allowContentExtraction", AllowContentExtraction.ToString());
    parameters.Add("allowModifyAnnotations", AllowModifyAnnotations.ToString());
    parameters.Add("printQuality", PrintQuality);
    // Convert dictionary of params to JSON
    string jsonPayload = JsonConvert.SerializeObject(parameters);

    try
    {
        // URL of "PDF Security" endpoint
        string url = "https://api.pdf.co/v1/pdf/security/add";

        // Execute POST request with JSON payload
        string response = webClient.UploadString(url, jsonPayload);

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

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

            // Download generated PDF file
            webClient.DownloadFile(resultFileUrl, DestinationFile);

            Console.WriteLine("Generated PDF file saved as \"{0}\" file.", DestinationFile);
        }
        else
        {
            Console.WriteLine(json["message"].ToString());
        }
    }
    catch (WebException e)
    {
        Console.WriteLine(e.ToString());
    }

    webClient.Dispose();


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

/// <summary>
/// Uploads file to the cloud and return URL of uploaded file to use in further API calls.
/// </summary>
/// <param name="file">Source file name (path).</param>
/// <returns>URL of uploaded file</returns>
```

```
        static string UploadFile(string file)
        {
            // Create standard .NET web client instance
            WebClient webClient = new WebClient();

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

            try
            {
                // 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(file)));

                // 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", file); // You can use UploadData() instead if your file is in byte[] o

                    return uploadedFileUrl;
                }
                else
                {
                    // Display service reported error
                    Console.WriteLine(json["message"].ToString());
                }
            }
            catch (Exception e)
            {
                Console.WriteLine(e);
                throw;
            }
            finally
            {
                webClient.Dispose();
            }

            return null;
        }
    }
}
```

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

ON-PREMISE OFFLINE SDK

60 Day Free Trial or Visit PDF.co Web API Home Page
Explore PDF.co Web API Documentation
Explore Samples
Sign Up for PDF.co Web API 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