www.bytescout.com

How to replace text with image from PDF in C# and ByteScout Cloud API Server

ByteScout Cloud API Server is API server that is ready to use and can be installed and deployed in less than 30 minutes on your own Windows server or server in a cloud. It can save data and files on your local server-based file storage or in Amazon AWS S3 storage. Data is processed solely on the API server and is powered by ByteScout engine, no cloud services or Internet connection is required for data processing..

FOR MORE INFORMATION AND FREE TRIAL:

Download Free Trial SDK (on-premise version)

Read more about **ByteScout Cloud API Server**

Explore API Documentation

Get Free Training for ByteScout Cloud API Server

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.co
  <Import Project="$(MSBuildExtensionsPath)\$(MSBuildToolsVersion)\Microsoft.Common.pr
  <PropertyGroup>
    <Configuration Condition=" '$(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)|$(Platform)' == 'Release|AnyCPU' ">
    <PlatformTarget>AnyCPU</PlatformTarget>
    <DebugType>pdbonly</DebugType>
    <Optimize>true</Optimize>
    <OutputPath>bin\Release\</OutputPath>
    <DefineConstants>TRACE</DefineConstants>
    <ErrorReport>prompt</ErrorReport>
    <WarningLevel>4</WarningLevel>
  </PropertyGroup>
  <ItemGroup>
    <Reference Include="Newtonsoft.Json, Version=10.0.0.0, Culture=neutral, PublicKeyT
      <HintPath>packages\Newtonsoft.Json.10.0.3\lib\net40\Newtonsoft.Json.dll</HintPat
      <Private>True</Private>
    </Reference>
    <Reference Include="System" />
    <Reference Include="System.Core" />
    <Reference Include="System.Xml.Linq" />
    <Reference Include="System.Data" />
    <Reference Include="System.Xml" />
  </ItemGroup>
  <ItemGroup>
    <Compile Include="Program.cs" />
    <Compile Include="Properties\AssemblyInfo.cs" />
  </ItemGroup>
  <ItemGroup>
    <None Include="packages.config" />
  </ItemGroup>
  <Import Project="$(MSBuildToolsPath)\Microsoft.CSharp.targets" />
  <!-- To modify your build process, add your task inside one of the targets below and
       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", "ByteSco
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
                {1E1C2C34-017E-4605-AE2B-55EA3313BE51}.Debug|Any CPU.Build.0 = Debug|A
                {1E1C2C34-017E-4605-AE2B-55EA3313BE51}.Release|Any CPU.ActiveCfg = Rel
                {1E1C2C34-017E-4605-AE2B-55EA3313BE51}.Release|Any CPU.Build.0 = Relea
        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;
using System.Collections.Generic;
using Newtonsoft.Json;

// Cloud API asynchronous "Replace Text With Image from PDF" job example.

namespace ByteScoutWebApiExample
```

```
{
        // Please NOTE: In this sample we're assuming Cloud Api Server is hosted at "h
        // If it's not then please replace this with with your hosting url.
        class Program
        {
                // The authentication key (API Key).
                // Get your own by registering at https://app.pdf.co/documentation/api
                const String API_KEY = "**********************************";

                // Direct URL of source PDF file.
                const string SourceFileUrl = "https://bytescout-com.s3.amazonaws.com/f
                // PDF document password. Leave empty for unprotected documents.
                const string Password = "";
                // Destination PDF file name
                const string DestinationFile = @".\result.pdf";
                // (!) 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);

                        // URL for `Replace Text With Image from PDF` API call
                        string url = "https://localhost/pdf/edit/replace-text-with-ima

                        // Prepare requests params as JSON
                        Dictionary<string, object> parameters = new Dictionary<string,
                        parameters.Add("name", Path.GetFileName(DestinationFile));
                        parameters.Add("password", Password);
                        parameters.Add("url", SourceFileUrl);
                        parameters.Add("async", Async);
                        parameters.Add("searchString", "/creativecommons.org/licenses/
                        parameters.Add("replaceImage", "https://bytescout-com.s3.amazo

                        // Convert dictionary of params to JSON
                        string jsonPayload = JsonConvert.SerializeObject(parameters);

                        try
                        {
                                // Execute POST request with JSON payload
                                string response = webClient.UploadString(url, jsonPayl

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

                                if (json["error"].ToObject<bool>() == false)
                                {
                                        // Asynchronous job ID
                                        string jobId = json["jobId"].ToString();
                                        // URL of generated PDF file that will availab
                                        string resultFileUrl = json["url"].ToString();

                                        // Check the job status in a loop.
                                        // If you don't want to pause the main thread
                                        // to use a separate thread for the status che
```

```
                                do
                                {
                                        string status = CheckJobStatus(jobId);

                                        // Display timestamp and status (for d
                                        Console.WriteLine(DateTime.Now.ToLongT

                                        if (status == "success")
                                        {
                                                // Download PDF file
                                                webClient.DownloadFile(resultF

                                                Console.WriteLine("Generated P
                                                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://localhost/job/check?jobid=" + jo

                        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>
```

---

visit www.ByteScout.com

visit www.PDF.co