www.bytescout.com

How to async file upload and async split PDF for PDF splitting API in VB.NET using PDF.co Web API

What is PDF.co Web API? It is 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.sln

```
Microsoft Visual Studio Solution File, Format Version 12.00
# Visual Studio 15
VisualStudioVersion = 15.0.26730.10
MinimumVisualStudioVersion = 10.0.40219.1
Project("{F184B08F-C81C-45F6-A57F-5ABD9991F28F}") = "ByteScoutWebApiExample", "ByteScoutWebApiExample
EndProject
Global
	GlobalSection(SolutionConfigurationPlatforms) = preSolution
		Debug|Any CPU = Debug|Any CPU
		Release|Any CPU = Release|Any CPU
	EndGlobalSection
	GlobalSection(ProjectConfigurationPlatforms) = postSolution
		{9B91124C-66C3-4BD9-B29E-168C1ABB15AC}.Debug|Any CPU.ActiveCfg = Debug|Any CPU
```

```
		{9B91124C-66C3-4BD9-B29E-168C1ABB15AC}.Debug|Any CPU.Build.0 = Debug|Any CPU
		{9B91124C-66C3-4BD9-B29E-168C1ABB15AC}.Release|Any CPU.ActiveCfg = Release|Any CPU
		{9B91124C-66C3-4BD9-B29E-168C1ABB15AC}.Release|Any CPU.Build.0 = Release|Any CPU
	EndGlobalSection
	GlobalSection(SolutionProperties) = preSolution
		HideSolutionNode = FALSE
	EndGlobalSection
	GlobalSection(ExtensibilityGlobals) = postSolution
		SolutionGuid = {4576C9BB-A42D-46A8-9198-7E2982E122FA}
	EndGlobalSection
EndGlobal
```

ByteScoutWebApiExample.vbproj

```
<?xml version="1.0" encoding="utf-8"?>
<Project ToolsVersion="15.0" 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>{9B91124C-66C3-4BD9-B29E-168C1ABB15AC}</ProjectGuid>
    <OutputType>Exe</OutputType>
    <StartupObject>ByteScoutWebApiExample.Module1</StartupObject>
    <RootNamespace>ByteScoutWebApiExample</RootNamespace>
    <AssemblyName>ByteScoutWebApiExample</AssemblyName>
    <FileAlignment>512</FileAlignment>
    <MyType>Console</MyType>
    <TargetFrameworkVersion>v4.0</TargetFrameworkVersion>
  </PropertyGroup>
  <PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Debug|AnyCPU' ">
    <PlatformTarget>AnyCPU</PlatformTarget>
    <DebugSymbols>true</DebugSymbols>
    <DebugType>full</DebugType>
    <DefineDebug>true</DefineDebug>
    <DefineTrace>true</DefineTrace>
    <OutputPath>bin\Debug\</OutputPath>
    <DocumentationFile>ByteScoutWebApiExample.xml</DocumentationFile>
    <NoWarn>42016,41999,42017,42018,42019,42032,42036,42020,42021,42022</NoWarn>
  </PropertyGroup>
  <PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Release|AnyCPU' ">
    <PlatformTarget>AnyCPU</PlatformTarget>
    <DebugType>pdbonly</DebugType>
    <DefineDebug>false</DefineDebug>
    <DefineTrace>true</DefineTrace>
    <Optimize>true</Optimize>
    <OutputPath>bin\Release\</OutputPath>
    <DocumentationFile>ByteScoutWebApiExample.xml</DocumentationFile>
    <NoWarn>42016,41999,42017,42018,42019,42032,42036,42020,42021,42022</NoWarn>
  </PropertyGroup>
  <PropertyGroup>
    <OptionExplicit>On</OptionExplicit>
  </PropertyGroup>
  <PropertyGroup>
    <OptionCompare>Binary</OptionCompare>
  </PropertyGroup>
  <PropertyGroup>
    <OptionStrict>Off</OptionStrict>
  </PropertyGroup>
  <PropertyGroup>
    <OptionInfer>On</OptionInfer>
  </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>
    </Reference>
    <Reference Include="System" />
```

```
    <Reference Include="System.Data" />
    <Reference Include="System.Deployment" />
    <Reference Include="System.Xml" />
    <Reference Include="System.Core" />
    <Reference Include="System.Xml.Linq" />
    <Reference Include="System.Data.DataSetExtensions" />
  </ItemGroup>
  <ItemGroup>
    <Import Include="Microsoft.VisualBasic" />
    <Import Include="System" />
    <Import Include="System.Collections" />
    <Import Include="System.Collections.Generic" />
    <Import Include="System.Data" />
    <Import Include="System.Diagnostics" />
    <Import Include="System.Linq" />
    <Import Include="System.Xml.Linq" />
  </ItemGroup>
  <ItemGroup>
    <Compile Include="Module1.vb" />
    <Compile Include="My Project\AssemblyInfo.vb" />
    <Compile Include="My Project\Application.Designer.vb">
      <AutoGen>True</AutoGen>
      <DependentUpon>Application.myapp</DependentUpon>
    </Compile>
    <Compile Include="My Project\Resources.Designer.vb">
      <AutoGen>True</AutoGen>
      <DesignTime>True</DesignTime>
      <DependentUpon>Resources.resx</DependentUpon>
    </Compile>
    <Compile Include="My Project\Settings.Designer.vb">
      <AutoGen>True</AutoGen>
      <DependentUpon>Settings.settings</DependentUpon>
      <DesignTimeSharedInput>True</DesignTimeSharedInput>
    </Compile>
  </ItemGroup>
  <ItemGroup>
    <EmbeddedResource Include="My Project\Resources.resx">
      <Generator>VbMyResourcesResXFileCodeGenerator</Generator>
      <LastGenOutput>Resources.Designer.vb</LastGenOutput>
      <CustomToolNamespace>My.Resources</CustomToolNamespace>
      <SubType>Designer</SubType>
    </EmbeddedResource>
  </ItemGroup>
  <ItemGroup>
    <None Include="My Project\Application.myapp">
      <Generator>MyApplicationCodeGenerator</Generator>
      <LastGenOutput>Application.Designer.vb</LastGenOutput>
    </None>
    <None Include="My Project\Settings.settings">
      <Generator>SettingsSingleFileGenerator</Generator>
      <CustomToolNamespace>My</CustomToolNamespace>
      <LastGenOutput>Settings.Designer.vb</LastGenOutput>
    </None>
    <None Include="packages.config" />
    <Content Include="sample.pdf">
      <CopyToOutputDirectory>Always</CopyToOutputDirectory>
    </Content>
  </ItemGroup>
  <Import Project="$(MSBuildToolsPath)\Microsoft.VisualBasic.targets" />
</Project>
```

Module1.vb

```
Imports System.IO
Imports System.Net
Imports System.Threading
Imports Newtonsoft.Json.Linq
```

```
Module Module1

    ' The authentication key (API Key).
    ' Get your own by registering at https://app.pdf.co/documentation/api
    Const API_KEY As String = "**********************************"

    ' Source PDF file to split
    Const SourceFile As String = ".\sample.pdf"
    ' Comma-separated list of page numbers (or ranges) to process. Example: '1,3-5,7-'.
    Const Pages As String = "1-2,3-"
    ' (!) Make asynchronous job
    Const Async As Boolean = True

    Sub Main()

        ' Create standard .NET web client instance
        Dim webClient As 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
        Dim query As String = 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
            Dim response As String = webClient.DownloadString(query)

            ' Parse JSON response
            Dim json As JObject = JObject.Parse(response)

            If json("error").ToObject(Of Boolean) = False Then

                ' Get URL to use for the file upload
                Dim uploadUrl As String = json("presignedUrl").ToString()
                ' Get URL of uploaded file to use with later API calls
                Dim uploadedFileUrl As String = 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 byte a

                ' 3. SPLIT UPLOADED PDF

                ' Prepare URL for `Split PDF` API call
                ' Prepare URL for `Split PDF` API call
                query = Uri.EscapeUriString(String.Format(
                    "https://api.pdf.co/v1/pdf/split?pages={0}&url={1}&async={2}",
                    Pages,
                    uploadedFileUrl,
                    Async))

                Try
                    ' Execute request
                    response = webClient.DownloadString(query)

                    ' Parse JSON response
                    json = JObject.Parse(response)

                    If json("error").ToObject(Of Boolean) = False Then

                        ' Asynchronous job ID
                        Dim jobId As String = json("jobId").ToString()
                        ' URL of generated JSON file available after the job completion; it will contain URLs of result PDF files.
                        Dim resultJsonFileUrl As String = 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
                            Dim status As String = CheckJobStatus(jobId) ' Possible statuses: "working", "failed", "aborted", "suc
```

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

                    If status = "success" Then

                        ' Download JSON file as string
                        Dim jsonFileString As String = webClient.DownloadString(resultJsonFileUrl)

                        Dim resultFilesUrls As JArray = JArray.Parse(jsonFileString)

                        ' Download generated PDF files
                        Dim part As Integer = 1
                        For Each token As JToken In resultFilesUrls

                            Dim resultFileUrl As String = token.ToString()
                            Dim localFileName As String = String.Format(".\part{0}.pdf", part)

                            webClient.DownloadFile(resultFileUrl, localFileName)

                            Console.WriteLine("Downloaded ""{0}"".", localFileName)
                            part = part + 1

                        Next

                        Exit Do

                    ElseIf status = "working" Then

                        ' Pause for a few seconds
                        Thread.Sleep(3000)

                    Else

                        Console.WriteLine(status)
                        Exit Do

                    End If

                Loop

            Else
                Console.WriteLine(json("message").ToString())
            End If

        Catch ex As WebException
            Console.WriteLine(ex.ToString())
        End Try

    End If

Catch ex As WebException
    Console.WriteLine(ex.ToString())
End Try

webClient.Dispose()


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

End Sub

''' <summary>
''' Check job status
''' </summary>
Function CheckJobStatus(jobId As String) As String

    Using webClient As WebClient = New WebClient()

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

        Dim url As String = "https://api.pdf.co/v1/job/check?jobid=" + jobId

        Dim response As String = webClient.DownloadString(url)
        Dim json As JObject = JObject.Parse(response)
```

```
            Return Convert.ToString(json("status"))
        End Using
    End Function

End Module
```

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

www.bytescout.com