www.bytescout.com

How to convert PDF to XLS from URL asynchronously for PDF to excel API in VB.NET with ByteScout Cloud API Server

Step By Step Instructions on how to convert PDF to XLS from URL asynchronously for PDF to excel API in VB.NET

This page displays the code samples for programming in VB.NET. ByteScout Cloud API Server was designed to assist PDF to excel API in VB.NET. ByteScout Cloud API Server is the ready to deploy Web API Server that can be deployed in less than thirty minutes into your own in-house Windows server (no Internet connnection is required to process data!) or into private cloud server. Can store data on in-house local server based storage or in Amazon AWS S3 bucket. Processing data solely on the server using built-in ByteScout powered engine, no cloud services are used to process your data!.

Want to learn quickly? These fast application programming interfaces of ByteScout Cloud API Server for VB.NET plus the instruction and the code below will help to learn how to convert PDF to XLS from URL asynchronously. Open your VB.NET project and simply copy & paste the code and then run your app! Check VB.NET sample code examples to see if they respond to your needs and requirements for the project.

Free! Free! Free! ByteScout free trial version is available for FREE download from our website. Programming tutorials along with source code samples are assembled.

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

Module1.vb

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


' Please NOTE: In this sample we're assuming Cloud Api Server is hosted at "https://lo
' If it's not then please replace this with with your hosting url.

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

Module Module1

        ' Direct URL of source PDF file.
        Const SourceFileUrl As String = "https://bytescout-com.s3.amazonaws.com/files/
```

```
' Comma-separated list of page indices (or ranges) to process. Leave empty for
Const Pages As String = ""
' PDF document password. Leave empty for unprotected documents.
Const Password As String = ""
' Destination XLS file name
Const DestinationFile As String = ".\result.xls"
' (!) Make asynchronous job
Const Async As Boolean = True


Sub Main()

	' Create standard .NET web client instance
	Dim webClient As WebClient = New WebClient()

	' Prepare URL for `PDF To XLS` API call
	Dim query As String = Uri.EscapeUriString(String.Format(
		"https://localhost/pdf/convert/to/xls?name={0}&password={1}&pag
		Path.GetFileName(DestinationFile),
		Password,
		Pages,
		SourceFileUrl,
		Async))

	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

			' Asynchronous job ID
			Dim jobId As String = json("jobId").ToString()
			' URL of generated XLS file that will available after t
			Dim resultFileUrl As String = json("url").ToString()

			' Check the job status in a loop.
			' If you don't want to pause the main thread you can re
			' to use a separate thread for the status checking and
			Do
				Dim status As String = CheckJobStatus(jobId) '

				' Display timestamp and status (for demo purpos
				Console.WriteLine(DateTime.Now.ToLongTimeString

				If status = "success" Then

					' Download XLS file
					webClient.DownloadFile(resultFileUrl, D

					Console.WriteLine("Generated XLS file s
					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

                webClient.Dispose()

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

        End Sub

        Function CheckJobStatus(jobId As String) As String

                Using webClient As WebClient = New WebClient()

                        Dim url As String = "https://localhost/job/check?jobid=" + jobI

                        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 ByteScout Cloud API Server Home Page
Explore ByteScout Cloud API Server Documentation
Explore Samples
Sign Up for ByteScout Cloud API Server 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