www.bytescout.com

How to convert PDF to HTML from uploaded file asynchronously for PDF to HTML API in Python with ByteScout Cloud API Server

Follow this simple tutorial to learn convert PDF to HTML from uploaded file asynchronously to have PDF to HTML API in Python

These simple tutorials explain the code material for beginners and advanced programmers who are using Python. ByteScout Cloud API Server was designed to assist PDF to HTML API in Python. ByteScout Cloud API Server is the ready to use Web API Server that can be deployed in less than 30 minutes into your own in-house server 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 buil-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 Python plus the instruction and the code below will help to learn how to convert PDF to HTML from uploaded file asynchronously. Follow the tutorial and copy - paste code for Python into your project's code editor. Writing Python application mostly includes various stages of the software development so even if the functionality works please check it with your data and the production environment.

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:

ConvertPdfToHtmlFromUploadedFileAsynchronously.py

```
""" Cloud API asynchronous "PDF To Text" job example.
    Allows to avoid timeout errors when processing huge or scanned PDF documents.
"""
import os
import requests # pip install requests
import time
import datetime

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

# Base URL for PDF.co Web API requests
BASE_URL = "https://localhost"

# Source PDF file
SourceFile = ".\\sample.pdf"
# Comma-separated list of page indices (or ranges) to process. Leave empty for all pages. Example: '0,2-5,7-'.
Pages = ""
# PDF document password. Leave empty for unprotected documents.
Password = ""
# Destination Html file name
DestinationFile = ".\\result.html"
# Set to $true to get simplified HTML without CSS. Default is the rich HTML keeping the document design.
PlainHtml = False
# Set to $true if your document has the column layout like a newspaper.
ColumnLayout = False
# (!) Make asynchronous job
Async = True


def main(args = None):
    uploadedFileUrl = uploadFile(SourceFile)
    if (uploadedFileUrl != None):
        convertPdfToHtml(uploadedFileUrl, DestinationFile)


def convertPdfToHtml(uploadedFileUrl, destinationFile):
    """Converts PDF To Html using PDF.co Web API"""

    # Prepare URL for 'PDF To Html' API request
    url = "{}/pdf/convert/to/html?async={}&name={}&password={}&pages={}&simple={}&columns={}&url={}".format(
        BASE_URL,
        Async,
        os.path.basename(destinationFile),
        Password,
        Pages,
        PlainHtml,
        ColumnLayout,
        uploadedFileUrl
    )

    # Execute request and get response as JSON
    response = requests.get(url, headers={  "content-type": "application/octet-stream" })
    if (response.status_code == 200):
        json = response.json()

        if json["error"] == False:
            # Asynchronous job ID
            jobId = json["jobId"]
            #  URL of the result file
            resultFileUrl = json["url"]

            # 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.
            while True:
                status = checkJobStatus(jobId) # Possible statuses: "working", "failed", "aborted", "success".

                # Display timestamp and status (for demo purposes)
```

```
            print(datetime.datetime.now().strftime("%H:%M.%S") + ": " + status)

            if status == "success":
                # Download result file
                r = requests.get(resultFileUrl, stream=True)
                if (r.status_code == 200):
                    with open(destinationFile, 'wb') as file:
                        for chunk in r:
                            file.write(chunk)
                    print(f"Result file saved as \"{destinationFile}\" file.")
                else:
                    print(f"Request error: {response.status_code} {response.reason}")
                break
            elif status == "working":
                # Pause for a few seconds
                time.sleep(3)
            else:
                print(status)
                break
        else:
            # Show service reported error
            print(json["message"])
    else:
        print(f"Request error: {response.status_code} {response.reason}")


def checkJobStatus(jobId):
    """Checks server job status"""

    url = f"{BASE_URL}/job/check?jobid={jobId}"

    response = requests.get(url)
    if (response.status_code == 200):
        json = response.json()
        return json["status"]
    else:
        print(f"Request error: {response.status_code} {response.reason}")

    return None


def uploadFile(fileName):
    """Uploads file to the cloud"""

    # 1. RETRIEVE PRESIGNED URL TO UPLOAD FILE.

    # Prepare URL for 'Get Presigned URL' API request
    url = "{}/file/upload/get-presigned-url?contenttype=application/octet-stream&name={}".format(
        BASE_URL, os.path.basename(fileName))

    # Execute request and get response as JSON
    response = requests.get(url)
    if (response.status_code == 200):
        json = response.json()

        if json["error"] == False:
            # URL to use for file upload
            uploadUrl = json["presignedUrl"]
            # URL for future reference
            uploadedFileUrl = json["url"]

            # 2. UPLOAD FILE TO CLOUD.
            with open(fileName, 'rb') as file:
                requests.put(uploadUrl, data=file, headers={  "content-type": "application/octet-stream" })

            return uploadedFileUrl
        else:
            # Show service reported error
            print(json["message"])
    else:
        print(f"Request error: {response.status_code} {response.reason}")

    return None


if __name__ == '__main__':
    main()
```

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