www.bytescout.com

# How to split PDF from uploaded file asynchronously for PDF splitting API in Python using ByteScout Cloud API Server

## Step-by-step tutorial:How to split PDF from uploaded file asynchronously to have PDF splitting API in Python

On this page, you will find sample source codes which show you how to handle a complex task, such as, PDF splitting API in Python. PDF splitting API in Python can be applied with ByteScout Cloud API Server. 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!.

Python code snippet like this for ByteScout Cloud API Server works best when you need to quickly implement PDF splitting API in your Python application. For implementation of this functionality, please copy and paste the code below into your app using code editor. Then compile and run your app. Use of ByteScout Cloud API Server in Python is also described in the documentation given along with the product.

Our website provides free trial version of ByteScout Cloud API Server that gives source code samples to assist with your Python project.

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:

SplitPDFFromUploadedFileAsynchronously.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 numbers (or ranges) to process. Example: '1,3-5,7-'.
Pages = "1-2,3-"
# (!) Make asynchronous job
Async = True

def main(args = None):
    uploadedFileUrl = uploadFile(SourceFile)
    if (uploadedFileUrl != None):
        splitPDF(uploadedFileUrl)


def splitPDF(uploadedFileUrl):
    """Split PDF using PDF.co Web API"""

    # Prepare URL for 'Split PDF' API request
    url = "{}/pdf/split?async={}&pages={}&url={}".format(
        BASE_URL,
        Async,
        Pages,
        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
            resultFilePlaceholder = 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":

                    resJsonImgFiles = requests.get(resultFilePlaceholder)

                    # Download generated PNG files
                    part = 1

                    for resultFileUrl in resJsonImgFiles.json():
                        # Download Result File
                        r = requests.get(resultFileUrl, stream=True)
```

```
                    localFileUrl = f"Page{part}.pdf"

                    if r.status_code == 200:
                        with open(localFileUrl, 'wb') as file:
                            for chunk in r:
                                file.write(chunk)
                        print(f"Result file saved as \"{localFileUrl}\" file.")
                    else:
                        print(f"Request error: {response.status_code} {response.reason}")

                    part = part + 1

                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