

[www.bytescout.com](http://www.bytescout.com)

## **How to split PDF by text from uploaded file asynchronously for PDF splitting API in Python with PDF.co Web API**

PDF.co Web API: the flexible Web API that includes full set of functions from e-signature requests to data extraction, OCR, images recognition, pdf splitting and pdf splitting. Can also generate barcodes and read barcodes from images, scans 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](http://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

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

# Base URL for PDF.co Web API requests
BASE_URL = "https://api.pdf.co/v1"

# Source PDF file
SourceFile = ".\\sample.pdf"
# Split by Text
SplitText = "invoice number"
# (!) 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 requests params as JSON
    # See documentation: https://apidocs.pdf.co
    parameters = {}
    parameters["async"] = Async
    parameters["searchString"] = SplitText
    parameters["url"] = uploadedFileUrl

    # Prepare URL for 'Split PDF By Text' API request
    url = "{}pdf/split2".format(BASE_URL)

    # Execute request and get response as JSON
    response = requests.post(url, data=parameters, headers={ "x-api-key": API_KEY })
    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

                # 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 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.re

            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, headers={ "x-api-key": API_KEY })
    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="

```

```
BASE_URL, os.path.basename(fileName))

# Execute request and get response as JSON
response = requests.get(url, headers={ "x-api-key": API_KEY })
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={ "x-api-key": API_KEY, "co

            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 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](http://www.bytescout.com)