

## How to merge PDF documents from uploaded files asynchronously for PDF merging API in Python using ByteScout Cloud API Server

How to merge PDF documents from uploaded files asynchronously in Python with easy ByteScout code samples to make PDF merging API. Step-by-step tutorial

This page displays the step-by-step instructions and algorithm of how to merge PDF documents from uploaded files asynchronously and how to apply it in your application. PDF merging 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 connection 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!.

This simple and easy to understand sample source code in Python for ByteScout Cloud API Server contains different functions and options you should do calling the API to implement PDF merging API. This sample code in Python is all you need. Just copy-paste it to the code editor, then add a reference to ByteScout Cloud API Server and you are ready to try it! Easy to understand tutorials are available along with installed ByteScout Cloud API Server if you'd like to learn more about the topic and the details of the API.

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

# Source Code Files:

## MergePDFDocumentsFromUploadedFileAsynchronously.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 files
SourceFile_1 = "\\sample1.pdf"
SourceFile_2 = "\\sample2.pdf"

# Destination PDF file name
DestinationFile = "\\result.pdf"

# (!) Make asynchronous job
Async = True

def main(args = None):
    UploadedFileUrl_1 = uploadFile(SourceFile_1)
    UploadedFileUrl_2 = uploadFile(SourceFile_2)

    if (UploadedFileUrl_1 != None and UploadedFileUrl_2 != None):
        uploadedFileUrls = "{}:{}".format(UploadedFileUrl_1, UploadedFileUrl_2)
        mergeFiles(uploadedFileUrls, DestinationFile)

def mergeFiles(uploadedFileUrls, destinationFile):
    """Perform Merge using PDF.co Web API"""

    # Prepare URL for 'Merge PDF' API request
    url = "{}/pdf/merge?async={}&name={}&url={}".format(
        BASE_URL,
        Async,
        os.path.basename(destinationFile),
        uploadedFileUrls
    )

    # 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](http://www.ByteScout.com)

[visit www.PDF.co](http://www.PDF.co)

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