www.bytescout.com

How to optimize PDF from URL asynchronously for PDF optimization API in Python with ByteScout Cloud API Server

Step-by-step tutorial:How to optimize PDF from URL asynchronously to have PDF optimization API in Python

Quick guide:Learn how to optimize PDF from URL asynchronously in Python. PDF optimization 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!.

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 optimization 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.

ByteScout Cloud API Server - free trial version is available on our website. Also, there are other code samples to help you with your Python application included into trial version.

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:

OptimizePdfFromUrlAsynchronously.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"

# Direct URL of source PDF file.
SourceFileURL = "https://bytescout-com.s3.amazonaws.com/files/demo-files/cloud-api/pdf-optimize/sample.pdf"
# PDF document password. Leave empty for unprotected documents.
Password = ""
# Destination PDF file name
DestinationFile = ".\\result.pdf"
# (!) Make asynchronous job
Async = True


def main(args = None):
    optimizePDF(SourceFileURL, DestinationFile)


def optimizePDF(uploadedFileUrl, destinationFile):
    """Optimize PDF using PDF.co Web API"""

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

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

VIDEO

https://www.youtube.com/watch?v=NEwNs2b9YN8

visit www.ByteScout.com

visit www.PDF.co