www.bytescout.com

How to parse from url for document parser API in Python and ByteScout Cloud API Server

How to parse from url in Python with easy ByteScout code samples to make document parser API. Step-by-step tutorial

Writing of the code to parse from url in Python can be done by developers of any level using ByteScout Cloud API Server. ByteScout Cloud API Server was designed to assist document parser API in Python. 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!.

Use the code displayed below in your application to save a lot of time on writing and testing code. Open your Python project and simply copy & paste the code and then run your app! You can use these Python sample examples in one or many applications.

Trial version of ByteScout is available for free download from our website. This and other source code samples for Python and other programming languages are available.

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:

MultiPageTable-template1.yml

```
---
# Template that demonstrates parsing of multi-page table using only
# regular expressions for the table start, end, and rows.
# If regular expression cannot be written for every table row (for example,
# if the table contains empty cells), try the second method demonstrated
# in 'MultiPageTable-template2.yml' template.
templateVersion: 2
templatePriority: 0
sourceId: Multipage Table Test
detectionRules:
  keywords:
  - Sample document with multi-page table
fields:
  total:
    expression: TOTAL {{DECIMAL}}
tables:
- name: table1
  start:
    # regular expression to find the table start in document
    expression: Item\s+Description\s+Price\s+Qty\s+Extended Price
  end:
    # regular expression to find the table end in document
    expression: TOTAL\s+\d+\.\d\d
  row:
    # regular expression to find table rows
    expression: '^\s*(?<itemNo>\d+)\s+(?<description>.+?)\s+(?<price>\d+\.\d\d)\s+(?<qty>\d+)\s+(?<extPrice>\d+\.\d\d
  columns:
  - name: itemNo
    type: integer
  - name: description
    type: string
  - name: price
    type: decimal
  - name: qty
    type: integer
  - name: extPrice
    type: decimal
  multipage: true
```

ParseFromUrl.py

```
import os
import requests # pip install requests

# 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 url
SourceFileUrl = "https://bytescout-com.s3.amazonaws.com/files/demo-files/cloud-api/document-parser/MultiPageTabl

# Destination JSON file name
DestinationFile = ".\\result.json"

# Template text. Use Document Parser SDK (https://bytescout.com/products/developer/documentparsersdk/index.htm
# to create templates.
# Read template from file:
```

```
file_read = open(".\\MultiPageTable-template1.yml", mode='r', encoding="utf-8",errors="ignore")
Template = file_read.read()
file_read.close()

def main(args = None):
    PerformDocumentParser(SourceFileUrl, Template, DestinationFile)

def PerformDocumentParser(uploadedFileUrl, template, destinationFile):

    # Content
    data = {
        'url': uploadedFileUrl,
        'template': template
    }

    # Prepare URL for 'Document Parser' API request
    url = "{}/pdf/documentparser".format(BASE_URL)

    # Execute request and get response as JSON
    response = requests.post(url, data= data)

    if (response.status_code == 200):
        json = response.json()

        if json["error"] == False:
            #  Get URL of result file
            resultFileUrl = json["url"]
            # 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}")
        else:
            # Show service reported error
            print(json["message"])
    else:
        print(f"Request error: {response.status_code} {response.reason}")


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