Mesclar arquivos PDF

Perguntei 12 anos, 5 meses atrás Modificado 28 dias atrás Visto 268 mil vezes

255

É possível, usando Python, mesclar arquivos PDF separados?

Supondo que sim, preciso estender isso um pouco mais. Espero percorrer as pastas em um diretório e repetir este procedimento.

E posso estar abusando da sorte, mas é possível excluir uma página que está contida em cada um dos PDFs (minha geração de relatório sempre cria uma página extra em branco).

Pitão pdf arquivo-io pypdf

Compartilhar Seguir

editado em 12 de outubro de 2021 às 1:37

Woody1193
6.524 4 39 75

perguntado em 9 de agosto de 2010 às 22h23

Btibert3
38k 44 127 167

15 respostas

Classificado por:

Pontuação mais alta (padrão)

419

Você pode usar a classe PyPdf2 s `PdfMerger` .

Concatenação de arquivos

Você pode simplesmente concatenar arquivos usando o `append` método.

```
from PyPDF2 import PdfMerger

pdfs = ['file1.pdf', 'file2.pdf', 'file3.pdf', 'file4.pdf']

merger = PdfMerger()

for pdf in pdfs:
    merger.append(pdf)

merger.write("result.pdf")
merger.close()
```

Você pode passar identificadores de arquivo em vez de caminhos de arquivo, se desejar.

Mesclagem de arquivos

Se você deseja um controle mais refinado da mesclagem, existe um `merge` método do

que você pode inserir as páginas em qualquer lugar do arquivo. O `append` método pode ser pensado como `merge` onde o ponto de inserção é o final do arquivo.

por exemplo

```
merger.merge(2, pdf)
```

Aqui inserimos todo o pdf na saída, mas na página 2.

Intervalos de páginas

Se você deseja controlar quais páginas são anexadas de um determinado arquivo, você pode usar o `pages` argumento de palavra -chave de `append` e `merge` , passando uma tupla no formulário `(start, stop[, step])` (como a função regular `range`).

por exemplo

```
merger.append(pdf, pages=(0, 3))    # first 3 pages
merger.append(pdf, pages=(0, 6, 2)) # pages 1,3, 5
```

Se você especificar um intervalo inválido, receberá um arquivo `IndexError` .

Nota: também para evitar que os arquivos sejam deixados abertos, o `PdfFileMerger` método s close deve ser chamado quando o arquivo mesclado for gravado. Isso garante que todos os arquivos sejam fechados (entrada e saída) em tempo hábil. É uma pena que `PdfFileMerger` não seja implementado como um gerenciador de contexto, então podemos usar a palavra-`with` chave, evitar a chamada de fechamento explícita e obter alguma segurança de exceção fácil.

Você também pode querer olhar o `pdfcat` script fornecido como parte do pypdf2. Você pode potencialmente evitar a necessidade de escrever código completamente.

O github PyPdf2 também inclui alguns códigos de exemplo que demonstram a mesclagem.

PyMuPdfGenericName

Outra biblioteca que talvez valha a pena dar uma olhada é PyMuPdf . A fusão é igualmente simples.

Da linha de comando:

```
python -m fitz join -o result.pdf file1.pdf file2.pdf file3.pdf
```

e do código

```
import fitz
```

```
for pdf in ['file1.pdf', 'file2.pdf', 'file3.pdf']:
    with fitz.open(pdf) as mfile:
        result.insertPDF(mfile)

result.save("result.pdf")
```

Com muitas opções, detalhadas no wiki de projetos .

Compartilhar Seguir

editado em 22 de maio de 2022 às 20:31

Martin Thoma
118k 152 590 910

respondido em 21 de junho de 2016 às 13h12

Paul Rooney
20,3k 9 41 61

5 O PyMuPDF foi muito mais rápido do que o PyPDF2 para o que eu tinha que fazer (Concatenar 280 PDFs de página única). Obrigado! – Skusku 7 de janeiro de 2022 às 11h34

2 PyPDF2 agora também é mantido novamente :-) Acabei de vincular à sua resposta: pypdf2.readthedocs.io/en/latest/user/merging-pdfs.html – Martin Thoma 9 de abril de 2022 às 13h58

1 pyPDF2 está funcionando bem, mas os links internos não estão funcionando, ou seja, não movendo para a seção especificada ao clicar. Qualquer ideia? Eu testei mesmo com um único arquivo, ainda o mesmo problema – Rami 11 de maio de 2022 às 17h20

Eu nunca tentei isso. Este é apenas um processo básico de fusão, pode ser necessário entrar e corrigir os links internos depois. Fico feliz em resolver isso, se você puder fornecer mais detalhes. Você pode até considerar fazer uma nova pergunta? – Paul Rooney 11 de maio de 2022 às 22h47

1 @Skusku O mesmo para mim, tive que combinar páginas de diferentes documentos PDF em um "one-pager". Com PyPDF2 (usando `merge_page()` e transformações), isso levou 12 minutos (!), com PyMuPDF apenas 2 segundos usando a abordagem aqui . – Splines 16 de junho de 2022 às 13h22

153

Use Pypdf ou seu sucessor PyPDF2 :

> Uma biblioteca Pure-Python construída como um kit de ferramentas PDF. É capaz de:
>
> - dividir documentos página por página,
> - mesclando documentos página por página,

(e muito mais)

Aqui está um exemplo de programa que funciona com ambas as versões.

```
#!/usr/bin/env python
import sys
try:
    from PyPDF2 import PdfFileReader, PdfFileWriter
except ImportError:
    from pyPdf import PdfFileReader, PdfFileWriter

def pdf_cat(input_files, output_stream):
```

```
        # the data isn't read from the input files until the write
        # operation. Thanks to
        # https://stackoverflow.com/questions/6773631/problem-with-closing-python-
pypdf-writing-getting-a-valueerror-i-o-operation/6773733#6773733
        for input_file in input_files:
            input_streams.append(open(input_file, 'rb'))
        writer = PdfFileWriter()
        for reader in map(PdfFileReader, input_streams):
            for n in range(reader.getNumPages()):
                writer.addPage(reader.getPage(n))
        writer.write(output_stream)
    finally:
        for f in input_streams:
            f.close()
        output_stream.close()

if __name__ == '__main__':
    if sys.platform == "win32":
        import os, msvcrt
        msvcrt.setmode(sys.stdout.fileno(), os.O_BINARY)
    pdf_cat(sys.argv[1:], sys.stdout)
```

Compartilhar Seguir

editado em 11 de junho de 2021 às 15h39

Yul Kang
429 4 9

respondido em 9 de agosto de 2010 às 22h40

Gilles 'Então, pare de ser mau'
102k 37 210 252

22 E agora, pypi.python.org/pypi/PyPDF2 , que é o projeto sucessor do PyPDF – David Fraser 22 de agosto de 2013 às 10:04

1 Funciona para mim apenas com abertura no modo binário (fluxos de entrada e também fluxo de saída). `open(input_file), 'r+b'` , e em vez de sys.stdout eu uso `output_stream = open('result.pdf', 'w+b')` . – Simeon Borko 23 de março de 2018 às 12h01

@SimeonBorko Drop the `+` , it means “read and write” and neither file is both read and written. I've added Windows support output support based on stackoverflow.com/questions/2374427/... – Gilles 'SO- stop being evil' Mar 23, 2018 at 18:20

PyPDF2/3 is not stable, how can I merge pdf files without PyPDF2/3 . – GoingMyWay Jun 19, 2019 at 3:03

2 I had to use `sys.stdout.buffer` using Python 3.6.8 (Linux) – Greyshack Aug 21, 2019 at 13:33

Merge all pdf files that are present in a dir

40

Put the pdf files in a dir. Launch the program. You get one pdf with all the pdfs merged.

```
import os
from PyPDF2 import PdfMerger

x = [a for a in os.listdir() if a.endswith(".pdf")]

merger = PdfMerger()
```

```
with open("result.pdf", "wb") as fout:
    merger.write(fout)
```

How would I make the same code above today

```
from glob import glob
from PyPDF2 import PdfMerger


def pdf_merge():
    ''' Merges all the pdf files in current directory '''
    merger = PdfMerger()
    allpdfs = [a for a in glob("*.pdf")]
    [merger.append(pdf) for pdf in allpdfs]
    with open("Merged_pdfs.pdf", "wb") as new_file:
        merger.write(new_file)


if __name__ == "__main__":
    pdf_merge()
```

Share Follow

edited Jan 4 at 10:24

answered Nov 17, 2017 at 17:40

PythonProgrammi
21.6k 3 39 34

1 I used this successfully – Merlin Apr 8, 2021 at 21:42

pyPDF2 is promising, but internal links are not working. Any idea? I tested even with single file, still same issue – Rami May 11, 2022 at 17:20

2 As of December 2022, I found that PdfFileMerger was depracated when I tried running the code in the second part. You need to replace PdfFileMerger with PdfMerger and it'll work just fine – R41nMak3R Dec 27, 2022 at 12:33

15

The `pdfrw` library can do this quite easily, assuming you don't need to preserve bookmarks and annotations, and your PDFs aren't encrypted. `cat.py` is an example concatenation script, and `subset.py` is an example page subsetting script.

The relevant part of the concatenation script -- assumes `inputs` is a list of input filenames, and `outfn` is an output file name:

```
from pdfrw import PdfReader, PdfWriter

writer = PdfWriter()
for inpfn in inputs:
    writer.addpages(PdfReader(inpfn).pages)
writer.write(outfn)
```

```
writer.addpages(PdfReader(inpfn).pages[:-1])
```

Disclaimer: I am the primary `pdfrw` author.

Share Follow

edited Sep 17, 2017 at 15:41

0 _
9,931 11 75 108

answered Apr 2, 2017 at 0:04

Patrick Maupin
7,935 2 22 42

2 This is the most stable one. – GoingMyWay Jun 19, 2019 at 3:14

3 This library deserves more reputation. – GoingMyWay Jun 19, 2019 at 3:20

I see internal links are not working, not navigating desired section. Any idea? – Rami May 12, 2022 at 9:44

1 i have to deal with some monster PDF files created by a client in autocad on a regular basis, and this library made much quicker work of them than the ones mentioned in other answers. highly highly recommend. – Rick Sep 12, 2022 at 15:56

Is it possible, using Python, to merge seperate PDF files?

9

Yes.

The following example merges all files in one folder to a single new PDF file:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

from argparse import ArgumentParser
from glob import glob
from pyPdf import PdfFileReader, PdfFileWriter
import os

def merge(path, output_filename):
    output = PdfFileWriter()

    for pdffile in glob(path + os.sep + '*.pdf'):
        if pdffile == output_filename:
            continue
        print("Parse '%s'" % pdffile)
        document = PdfFileReader(open(pdffile, 'rb'))
        for i in range(document.getNumPages()):
            output.addPage(document.getPage(i))

    print("Start writing '%s'" % output_filename)
    with open(output_filename, "wb") as f:
        output.write(f)

if __name__ == "__main__":
    parser = ArgumentParser()

    # Add more options if you like
    parser.add_argument("-o", "--output",
                        dest="output_filename"
```

```
    parser.add_argument("-p", "--path",
                        dest="path",
                        default=".",
                        help="path of source PDF files")

    args = parser.parse_args()
    merge(args.path, args.output_filename)
```

Share Follow

edited Oct 7, 2015 at 13:44

answered Mar 31, 2014 at 16:41

Martin Thoma
118k 152 590 910

3

```
from PyPDF2 import PdfFileMerger
import webbrowser
import os
dir_path = os.path.dirname(os.path.realpath(__file__))

def list_files(directory, extension):
    return (f for f in os.listdir(directory) if f.endswith('.' + extension))

pdfs = list_files(dir_path, "pdf")

merger = PdfFileMerger()

for pdf in pdfs:
    merger.append(open(pdf, 'rb'))

with open('result.pdf', 'wb') as fout:
    merger.write(fout)

webbrowser.open_new('file://'+ dir_path + '/result.pdf')
```

Git Repo: https://github.com/mahaguru24/Python_Merge_PDF.git

Share Follow

answered Jul 27, 2018 at 2:24

guruprasad mulay
73 1 12

3

You can use pikepdf too (source code documentation).

Example code could be (taken from the documentation):

```
from glob import glob

from pikepdf import Pdf

pdf = Pdf.new()

for file in glob('*.pdf'):  # you can change this to browse directories recursively
    with Pdf.open(file) as src:
        pdf.pages.extend(src.pages)
```

If you want to exclude pages, you might proceed another way, for instance copying pages to a new pdf (you can select which ones you do not copy, then, the `pdf.pages` object behaving like a list).

It is still actively maintained, which, as of february 2022, does not seem to be the case of PyPDF2 nor pdfrw.

I haven't benchmarked it, so I don't know if it is quicker or slower than other solutions.

One advantage over PyMuPDF, in my case, is that an official Ubuntu package is available (python3-pikepdf), what is practical to package my own software depending on it.

Share Follow

answered Feb 22, 2022 at 18:01

zezollo
4,397 4 27 57

---

here, http://pieceofpy.com/2009/03/05/concatenating-pdf-with-python/, gives an solution.

3

similarly:

```
from pyPdf import PdfFileWriter, PdfFileReader

def append_pdf(input,output):
    [output.addPage(input.getPage(page_num)) for page_num in range(input.numPages)]

output = PdfFileWriter()

append_pdf(PdfFileReader(file("C:\\sample.pdf","rb")),output)
append_pdf(PdfFileReader(file("c:\\sample1.pdf","rb")),output)
append_pdf(PdfFileReader(file("c:\\sample2.pdf","rb")),output)
append_pdf(PdfFileReader(file("c:\\sample3.pdf","rb")),output)

output.write(file("c:\\combined.pdf","wb"))
```

------ Updated on 25th Nov. ------

------ Seems above code doesn't work anymore------

------ Please use the following:------

```
from PyPDF2 import PdfFileMerger, PdfFileReader
import os

merger = PdfFileMerger()

file_folder = "C:\\My Ducoments\\"

root, dirs, files = next(os.walk(file_folder))

for path, subdirs, files in os.walk(root):
    for f in files:
```

```
merger.write(file_folder + "Economists-1.pdf")
```

Share Follow

edited Nov 25, 2022 at 1:43

answered Jul 18, 2014 at 9:27

Mark K
8,155 13 54 110

what is `file` ? – Hammad Nov 24, 2022 at 18:56

1 @Hammad, thanks for the comment. Seems the old code doesn't work anymore. I've updated the answer with valid codes. :) – Mark K Nov 25, 2022 at 1:44

2

Here's a time comparison for the most common answers for my specific use case: combining a list of 5 large single-page pdf files. I ran each test twice.

(Disclaimer: I ran this function within Flask, your mileage may vary)

TL;DR

`pdfrw` is the fastest library for combining pdfs out of the 3 I tested.

PyPDF2

```
start = time.time()
merger = PdfFileMerger()
for pdf in all_pdf_obj:
    merger.append(
        os.path.join(
            os.getcwd(), pdf.filename # full path
                )
            )
formatted_name = f'Summary_Invoice_{date.today()}.pdf'
merge_file = os.path.join(os.getcwd(), formatted_name)
merger.write(merge_file)
merger.close()
end = time.time()
print(end - start) #1 66.50084733963013 #2 68.2995400428772
```

PyMuPDF

```
start = time.time()
result = fitz.open()

for pdf in all_pdf_obj:
    with fitz.open(os.path.join(os.getcwd(), pdf.filename)) as mfile:
        result.insertPDF(mfile)
formatted_name = f'Summary_Invoice_{date.today()}.pdf'

result.save(formatted_name)
end = time.time()
```

pdfrw

```
start = time.time()
result = fitz.open()

writer = PdfWriter()
for pdf in all_pdf_obj:
    writer.addpages(PdfReader(os.path.join(os.getcwd(), pdf.filename)).pages)

formatted_name = f'Summary_Invoice_{date.today()}.pdf'
writer.write(formatted_name)
end = time.time()
print(end - start) #1 0.6040127277374268 #2 0.9576816558837891
```

Share Follow

answered Jul 30, 2021 at 5:31

koopmac
893 8 24

1

A slight variation using a dictionary for greater flexibility (e.g. sort, dedup):

```
import os
from PyPDF2 import PdfFileMerger
# use dict to sort by filepath or filename
file_dict = {}
for subdir, dirs, files in os.walk("<dir>"):
    for file in files:
        filepath = subdir + os.sep + file
        # you can have multiple endswith
        if filepath.endswith((".pdf", ".PDF")):
            file_dict[file] = filepath
# use strict = False to ignore PdfReadError: Illegal character error
merger = PdfFileMerger(strict=False)

for k, v in file_dict.items():
    print(k, v)
    merger.append(v)

merger.write("combined_result.pdf")
```

Share Follow

answered Feb 19, 2019 at 22:40

Ogaga Uzoh
1,957 1 9 12

1

I used pdf unite on the linux terminal by leveraging subprocess (assumes one.pdf and two.pdf exist on the directory) and the aim is to merge them to three.pdf

```
import subprocess
subprocess.call(['pdfunite one.pdf two.pdf three.pdf'],shell=True)
```

Share Follow

answered Feb 1, 2020 at 0:54

1 This would work, however calling subprocess is not preferable over using PdfFileMerger from PyPDF2. Using shell=true introduces a security hazard. – Cloudkollektiv Nov 16, 2020 at 14:36

1

You can use `PdfFileMerger` from the PyPDF2 module.

For example, to merge multiple PDF files from a list of paths you can use the following function:

```
from PyPDF2 import PdfFileMerger

# pass the path of the output final file.pdf and the list of paths
def merge_pdf(out_path: str, extracted_files: list [str]):
    merger   = PdfFileMerger()

    for pdf in extracted_files:
        merger.append(pdf)

    merger.write(out_path)
    merger.close()

merge_pdf('./final.pdf', extracted_files)
```

And this function to get all the files recursively from a parent folder:

```
import os

# pass the path of the parent_folder
def fetch_all_files(parent_folder: str):
    target_files = []
    for path, subdirs, files in os.walk(parent_folder):
        for name in files:
            target_files.append(os.path.join(path, name))
    return target_files

# get a list of all the paths of the pdf
extracted_files = fetch_all_files('./parent_folder')
```

Finally, you use the two functions declaring.a `parent_folder_path` that can contain multiple documents, and an `output_pdf_path` for the destination of the merged PDF:

```
# get a list of all the paths of the pdf
parent_folder_path = './parent_folder'
outup_pdf_path     = './final.pdf'

extracted_files = fetch_all_files(parent_folder_path)
merge_pdf(outup_pdf_path, extracted_files)
```

You can get the full code from here (Source): How to merge PDF documents using Python

Share Follow

answered Nov 13, 2021 at 18:43

Domenico Ruggiano
469 3 13

```
import os
from PyPDF2 import PdfFileMerger

def merge_pdfs(export_dir, input_dir, folder):
    current_dir = os.path.join(input_dir, folder)
    pdfs = os.listdir(current_dir)

    merger = PdfFileMerger()
    for pdf in pdfs:
        merger.append(open(os.path.join(current_dir, pdf), 'rb'))

    with open(os.path.join(export_dir, folder + ".pdf"), "wb") as fout:
        merger.write(fout)

export_dir = r"E:\Output"
input_dir = r"E:\Input"
folders = os.listdir(input_dir)
[merge_pdfs(export_dir, input_dir, folder) for folder in folders];
```

Share Follow

answered Mar 24, 2021 at 15:39

faysou
1,132 10 24

Use right python interpreter:

0

```
conda activate py_envs

pip install PyPDF2
```

Python code:

```
from PyPDF2 import PdfMerger

#set path files
import os
os.chdir('/ur/path/to/folder/')
cwd = os.path.abspath('')
files = os.listdir(cwd)

def merge_pdf_files():
    merger = PdfMerger()
    pdf_files = [x for x in files if x.endswith(".pdf")]
    [merger.append(pdf) for pdf in pdf_files]
    with open("merged_pdf_all.pdf", "wb") as new_file:
        merger.write(new_file)

if __name__ == "__main__":
    merge_pdf_files()
```

Share Follow

edited Jan 3 at 5:06

Azhar Khan
3,776 6 26 32

answered Nov 28, 2022 at 9:28

Kon Li
11 2

-1

```
import logging
logging.basicConfig(filename = 'output.log', level = logging.DEBUG, format = '%
(asctime)s %(levelname)s %(message)s' )

try:
    import glob, os
    import PyPDF2

    os.chdir(path)

    pdfs = []

    for file in glob.glob("*.pdf"):
        pdfs.append(file)

    if len(pdfs) == 0:
        logging.info("No pdf in the given directory")

    else:
        merger = PyPDF2.PdfFileMerger()

        for pdf in pdfs:
            merger.append(pdf)

        merger.write('result.pdf')
        merger.close()

except Exception as e:
    logging.error('Error has happened')
    logging.exception('Exception occured' + str(e))
```

Compartilhar Seguir

respondido em 28 de junho de 2022 às 3:28

Mathujan Sivananthan
19 1