



# Working with PDF files in Python

Last Updated : 30 Sep, 2024

---

All of you must be familiar with what PDFs are. In fact, they are one of the most important and widely used digital media. PDF stands for **Portable Document Format**. It uses **.pdf** extension. It is used to present and exchange documents reliably, independent of software, hardware, or operating system. Invented by **Adobe**, PDF is now an open standard maintained by the International Organization for Standardization (ISO). PDFs can contain links and buttons, form fields, audio, video, and business logic.

In this article, we will learn, how we can do various operations like:

- Extracting text from PDF
- Rotating PDF pages
- Merging PDFs
- Splitting PDF
- Adding watermark to PDF pages

**Installation:** Using simple python scripts!

We will be using a third-party module, pypdf.

[pypdf](#) is a python library built as a PDF toolkit. It is capable of:

- Extracting document information (title, author, ...)
- Splitting documents page by page
- Merging documents page by page
- Cropping pages
- Merging multiple pages into a single page
- Encrypting and decrypting PDF files
- and more!

To install pypdf, run the following command from the command line:

```
pip install pypdf
```

This module name is case-sensitive, so make sure the **y** is lowercase and everything else is uppercase. All the code and PDF files used in this tutorial/article are available [here](#).

## 1. Extracting text from PDF file

Python



```
# importing required classes
from pypdf import PdfReader

# creating a pdf reader object
reader = PdfReader('example.pdf')

# printing number of pages in pdf file
print(len(reader.pages))

# creating a page object
page = reader.pages[0]

# extracting text from page
print(page.extract_text())
```

The output of the above program looks like this:

```
20
```

```
PythonBasics
```

```
S.R.Doty
```

```
August27,2008
```

```
Contents
```

```
1Preliminaries
```

```
4
```

```
1.1WhatIsPython?.....
```

```
..4
```

```
1.2Installationanddocumentation.....
```

```
.....4 [and some more lines...]
```

Let us try to understand the above code in chunks:

```
reader = PdfReader('example.pdf')
```

- Here, we create an object of **PdfReader** class of pypdf module and pass the path to the PDF file & get a PDF reader object.

```
print(len(reader.pages))
```

- **pages** property gives the number of pages in the PDF file. For example, in our case, it is 20 (see first line of output).

```
pageObj = reader.pages[0]
```

- Now, we create an object of **PageObject** class of pypdf module. PDF reader object has function **pages[]** which takes page number (starting from index 0) as argument and returns the page object.

```
print(pageObj.extract_text())
```

- Page object has function **extract\_text()** to extract text from the PDF page.

**Note:** While PDF files are great for laying out text in a way that's easy for people to print and read, they're not straightforward for software to parse into plaintext. As such, pypdf might make mistakes when extracting text from a PDF and may even be unable to open some PDFs at all. It isn't much you can do about this, unfortunately. pypdf may simply be unable to work with some of your particular PDF files.

## 2. Rotating PDF pages

Python

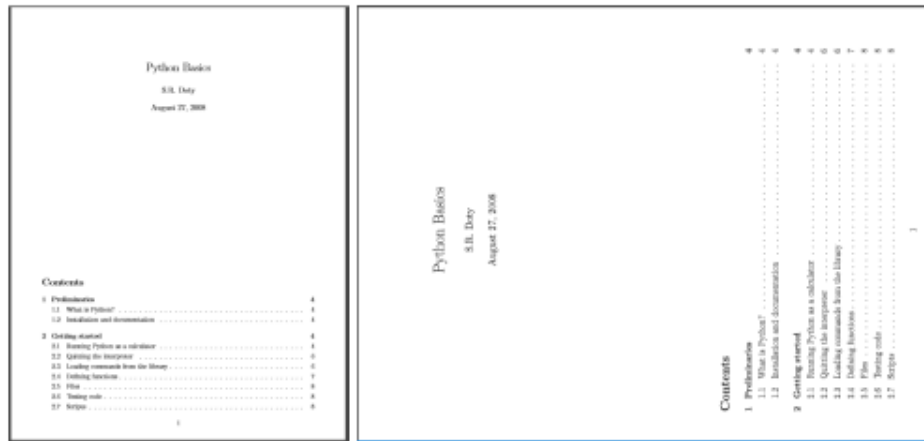


```
# importing the required classes
2 from pypdf import PdfReader, PdfWriter
3
4 def PDFrotate(origFileName, newFileName, rotation):
5
6     # creating a pdf Reader object
7     reader = PdfReader(origFileName)
```

[Python Basics](#) [Interview Questions](#) [Python Quiz](#) [Popular Packages](#) [Python Projects](#) [Practice Python](#) [AI Wit](#)

```
9     # creating a pdf writer object for new pdf
10    writer = PdfWriter()
11
12    # rotating each page
13    for page in range(len(reader.pages)):
14
15        pageObj = reader.pages[page]
16        pageObj.rotate(rotation)
17
18        # Add the rotated page object to the PDF writer
19        writer.add_page(pageObj)
20
21    # Write the rotated pages to the new PDF file
22    with open(newFileName, 'wb') as newFile:
23        writer.write(newFile)
24
25
26
27 def main():
28
29     # original pdf file name
30     origFileName = 'example.pdf'
31
32     # new pdf file name
33     newFileName = 'rotated_example.pdf'
34
35     # rotation angle
36     rotation = 270
37
38     # calling the PDFrotate function
39     PDFrotate(origFileName, newFileName, rotation)
40
41 if __name__ == "__main__":
42     # calling the main function
43     main()
```

Here, you can see how the first page of **rotated\_example.pdf** looks like ( right image) after rotation:



Some important points related to the above code:

- For rotation, we first create a PDF reader object of the original PDF.

```
writer = PdfWriter()
```

- Rotated pages will be written to a new PDF. For writing to PDFs, we use the object of **PdfWriter** class of pypdf module.

```
for page in range(len(pdfReader.pages)):
    pageObj = pdfReader.pages[page]
    pageObj.rotate(rotation)
    writer.add_page(pageObj)
```

- Now, we iterate each page of the original PDF. We get page object by **.pages[]** method of PDF reader class. Now, we rotate the page by **rotate()** method of page object class. Then, we add a page to PDF writer object using **addpage()** method of PDF writer class by passing the rotated page object.

```
newFile = open(newFileName, 'wb')
writer.write(newFile)
```

```
newFile.close()
```

- Now, we have to write the PDF pages to a new PDF file. Firstly, we open the new file object and write PDF pages to it using **write()** method of PDF writer object. Finally, we close the original PDF file object and the new file object.

### 3. Merging PDF files

#### Python

```
1  # importing required modules
2  from pypdf import PdfWriter
3
4
5  def PDFmerge(pdf, output):
6      # creating pdf file writer object
7      pdfWriter = PdfWriter()
8
9      # appending pdfs one by one
10     for pdf in pdf:
11         pdfWriter.append(pdf)
12
13     # writing combined pdf to output pdf file
14     with open(output, 'wb') as f:
15         pdfWriter.write(f)
16
17
18  def main():
19      # pdf files to merge
20      pdfs = ['example.pdf', 'rotated_example.pdf']
21
22      # output pdf file name
23      output = 'combined_example.pdf'
24
25      # calling pdf merge function
26      PDFmerge(pdfs, output)
27
28
29  if __name__ == "__main__":
30      # calling the main function
```

```
main()
```

The output of the above program is a combined PDF, **combined\_example.pdf**, obtained by merging **example.pdf** and **rotated\_example.pdf**.

- Let us have a look at important aspects of this program:

```
pdfWriter = PdfWriter()
```

- For merging, we use a pre-built class, **PdfWriter** of **pypdf** module. Here, we create an object **pdfwriter** of PDF writer class

```
# appending pdfs one by one
for pdf in pdfs:
    pdfWriter.append(pdf)
```

- Now, we append file object of each PDF to PDF writer object using the **append()** method.

```
# writing combined pdf to output pdf file
with open(output, 'wb') as f:
    pdfWriter.write(f)
```

- Finally, we write the PDF pages to the output PDF file using **write** method of PDF writer object.

#### 4. Splitting PDF file

Python

```
1 # importing the required modules
2 from pypdf import PdfReader, PdfWriter
3
4 def PDFsplit(pdf, splits):
5     # creating pdf reader object
6     reader = PdfReader(pdf)
```

```
8     # starting index of first slice
9     start = 0
10
11     # starting index of last slice
12     end = splits[0]
13
14
15     for i in range(len(splits)+1):
16         # creating pdf writer object for (i+1)th split
17         writer = PdfWriter()
18
19         # output pdf file name
20         outputpdf = pdf.split('.pdf')[0] + str(i) + '.pdf'
21
22         # adding pages to pdf writer object
23         for page in range(start,end):
24             writer.add_page(reader.pages[page])
25
26         # writing split pdf pages to pdf file
27         with open(outputpdf, "wb") as f:
28             writer.write(f)
29
30         # interchanging page split start position for
next split
31         start = end
32         try:
33             # setting split end position for next split
34             end = splits[i+1]
35         except IndexError:
36             # setting split end position for last split
37             end = len(reader.pages)
38
39
40     def main():
41         # pdf file to split
42         pdf = 'example.pdf'
43
44         # split page positions
45         splits = [2,4]
46
47         # calling PDFsplit function to split pdf
48         PDFsplit(pdf, splits)
```



```
50 if __name__ == "__main__":
51     # calling the main function
52     main()
```

Output will be three new PDF files with **split 1 (page 0,1)**, **split 2(page 2,3)**, **split 3(page 4-end)**.

No new function or class has been used in the above python program. Using simple logic and iterations, we created the splits of passed PDF according to the passed list **splits**.

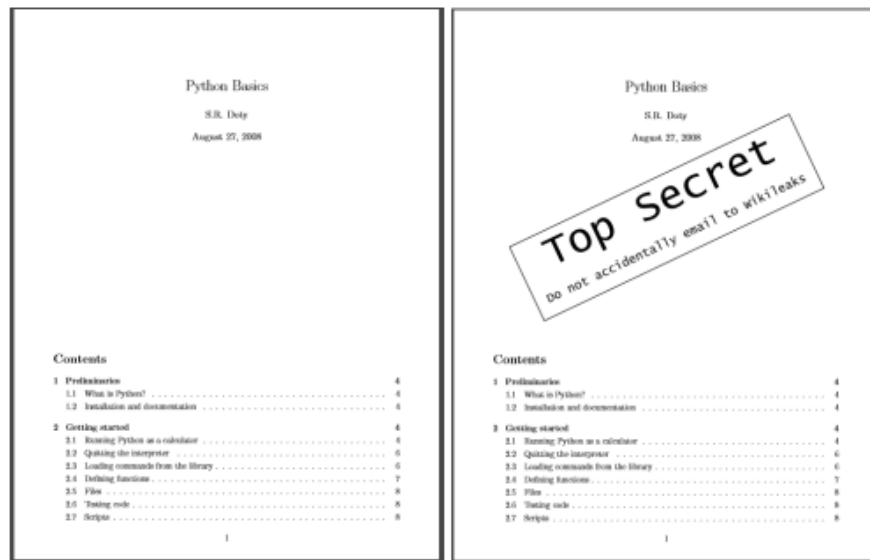
## 5. Adding watermark to PDF pages

### Python

```
1 # importing the required modules
2 from pypdf import PdfReader, PdfWriter
3
4 def add_watermark(wmFile, pageObj):
5     # creating pdf reader object of watermark pdf file
6     reader = PdfReader(wmFile)
7
8     # merging watermark pdf's first page with passed page
    object.
9     pageObj.merge_page(reader.pages[0])
10
11     # returning watermarked page object
12     return pageObj
13
14 def main():
15     # watermark pdf file name
16     mywatermark = 'watermark.pdf'
17
18     # original pdf file name
19     origFileName = 'example.pdf'
20
21     # new pdf file name
22     newFileName = 'watermarked_example.pdf'
23
24     # creating pdf File object of original pdf
25     pdfFileObj = open(origFileName, 'rb')
```

```
27     # creating a pdf Reader object
28     reader = PdfReader(pdfFileObj)
29
30     # creating a pdf writer object for new pdf
31     writer = PdfWriter()
32
33     # adding watermark to each page
34     for page in range(len(reader.pages)):
35         # creating watermarked page object
36         wmpageObj = add_watermark(mywatermark,
reader.pages[page])
37
38         # adding watermarked page object to pdf writer
39         writer.add_page(wmpageObj)
40
41     # writing watermarked pages to new file
42     with open(newFileName, 'wb') as newFile:
43         writer.write(newFile)
44
45     # closing the original pdf file object
46     pdfFileObj.close()
47
48 if __name__ == "__main__":
49     # calling the main function
50     main()
```

Here is how the first page of original (left) and watermarked (right) PDF file looks like:



- All the process is same as the page rotation example. Only difference is:

```
wmpageObj = add_watermark(mywatermark, pdfReader.pages[page])
```

- Page object is converted to watermarked page object using **add\_watermark()** function.
- Let us try to understand **add\_watermark()** function:

```
reader = PdfReader(wmFile)
pageObj.merge_page(reader.pages[0])
return pageObj
```

- Foremost, we create a PDF reader object of **watermark.pdf**. To the passed page object, we use **merge\_page()** function and pass the page object of the first page of the watermark PDF reader object. This will overlay the watermark over the passed page object.

And here we reach the end of this long tutorial on working with PDF files in python.

Now, you can easily create your own PDF manager!

**References:**

- <https://automatetheboringstuff.com/chapter13/>
- <https://pypi.org/project/pypdf/>

If you like GeeksforGeeks and would like to contribute, you can also write an article using [write.geeksforgeeks.org](https://write.geeksforgeeks.org) or mail your article to [review-team@geeksforgeeks.org](mailto:review-team@geeksforgeeks.org). See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or if you want to share more information about the topic discussed above.

If you have better suggestions about the products/services/tools/brands listed above or feel like something missing, please [Contact Us](#) and share your suggestions.

**N** Nikhil Kumar

28

### Previous Article

Python | Working with Pandas and  
XlsxWriter | Set – 3

### Next Article

## Similar Reads

### How to Crack PDF Files in Python?

Prerequisite: tqdm In this article, we will learn How to Crack a Protected PDF File Using Python. Here we will use the Brute force Method, to Crack a PDF File usin...

2 min read

### Working with zip files in Python

This article explains how one can perform various operations on a zip file using a simple python program. What is a zip file? ZIP is an archive file format that...

5 min read

## Working with large CSV files in Python

Data plays a key role in building machine learning and the AI model. In today's world where data is being generated at an astronomical rate by every computin...

4 min read

## Working with wav files in Python using Pydub

Audio files are a widespread means of transferring information. So let's see how to work with audio files using Python. Python provides a module called pydub t...

3 min read

## Working with Excel files in Python using Xlwings

Xlwings is a Python library that makes it easy to call Python from Excel and vice versa. It creates reading and writing to and from Excel using Python easily. It ca...

3 min read

## Working with csv files in Python

Python is one of the important fields for data scientists and many programmers to handle a variety of data. CSV (Comma-Separated Values) is one of the prevalent...

10 min read

## Send PDF File through Email using pdf-mail module

pdf\_mail module is that library of Python which helps you to send pdf documents through your Gmail account. Installing Library This module does not come built-...

2 min read

## Interact with PDF with PDF ChatBot

PDFs are widely used for sharing and viewing documents across various platforms and devices. Working with PDFs can sometimes be difficult and time-...

5 min read

## How to merge multiple excel files into a single files with Python ?

Normally, we're working with Excel files, and we surely have come across a scenario where we need to merge multiple Excel files into one. The traditional...

4 min read

## How to Scrape all PDF files in a Website?

Prerequisites: Implementing Web Scraping in Python with BeautifulSoup Web Scraping is a method of extracting data from the website and use that data for...

4 min read

Article Tags : [GBlog](#) [Python](#) [Listicles](#) [python](#)

Practice Tags : [python](#) [python](#)



Corporate & Communications Address:-  
A-143, 9th Floor, Sovereign Corporate  
Tower, Sector- 136, Noida, Uttar Pradesh  
(201305) | Registered Address:- K 061,  
Tower K, Gulshan Vivante Apartment,  
Sector 137, Noida, Gautam Buddh  
Nagar, Uttar Pradesh, 201305



### Company

[About Us](#)  
[Legal](#)  
[In Media](#)  
[Contact Us](#)  
[Advertise with us](#)  
[GFG Corporate Solution](#)  
[Placement Training Program](#)  
[GeeksforGeeks Community](#)

### DSA

[Data Structures](#)  
[Algorithms](#)  
[DSA for Beginners](#)

### Languages

[Python](#)  
[Java](#)  
[C++](#)  
[PHP](#)  
[GoLang](#)  
[SQL](#)  
[R Language](#)  
[Android Tutorial](#)  
[Tutorials Archive](#)

### Data Science & ML

[Data Science With Python](#)  
[Data Science For Beginner](#)  
[Machine Learning](#)

Basic DSA Problems  
DSA Roadmap  
Top 100 DSA Interview Problems  
DSA Roadmap by Sandeep Jain  
All Cheat Sheets

ML Maths  
Data Visualisation  
Pandas  
NumPy  
NLP  
Deep Learning

## Web Technologies

HTML  
CSS  
JavaScript  
TypeScript  
ReactJS  
NextJS  
Bootstrap  
Web Design

## Python Tutorial

Python Programming Examples  
Python Projects  
Python Tkinter  
Web Scraping  
OpenCV Tutorial  
Python Interview Question  
Django

## Computer Science

Operating Systems  
Computer Network  
Database Management System  
Software Engineering  
Digital Logic Design  
Engineering Maths  
Software Development  
Software Testing

## DevOps

Git  
Linux  
AWS  
Docker  
Kubernetes  
Azure  
GCP  
DevOps Roadmap

## System Design

High Level Design  
Low Level Design  
UML Diagrams  
Interview Guide  
Design Patterns  
OOAD  
System Design Bootcamp  
Interview Questions

## Interview Preparation

Competitive Programming  
Top DS or Algo for CP  
Company-Wise Recruitment Process  
Company-Wise Preparation  
Aptitude Preparation  
Puzzles

## School Subjects

Mathematics  
Physics  
Chemistry  
Biology  
Social Science  
English Grammar  
Commerce  
World GK

## GeeksforGeeks Videos

DSA  
Python  
Java  
C++  
Web Development  
Data Science  
CS Subjects

