

How to Extract Text from PDF

Learn to use Python to extract text from PDFs



Costas Andreou

Apr 12 · 4 min read ★



Photo by Carl Heyerdahl on Unsplash

In this blog, we are going to examine the most popular libraries for processing PDFs with Python. A lot of information is shared in the form of PDF, and often we need to extract some details for further processing.

To assist it in my research in identifying the most popular python libraries, I looked across StackOverflow, Reddit and generally lots of google searches. I identified

numerous packages, each with its own strengths and weakness. Specifically, users across the internet seem to be using: PyPDF2, Textract, tika, pdfPlumber, pdfMiner.

During my research, however, for one reason or another, I was only able to get 3 of these libraries to work as expected. For some of these libraries, the set up was too complicated (missing dependencies, strange error messages, etc.)

Let us quickly review all these libraries anyway.

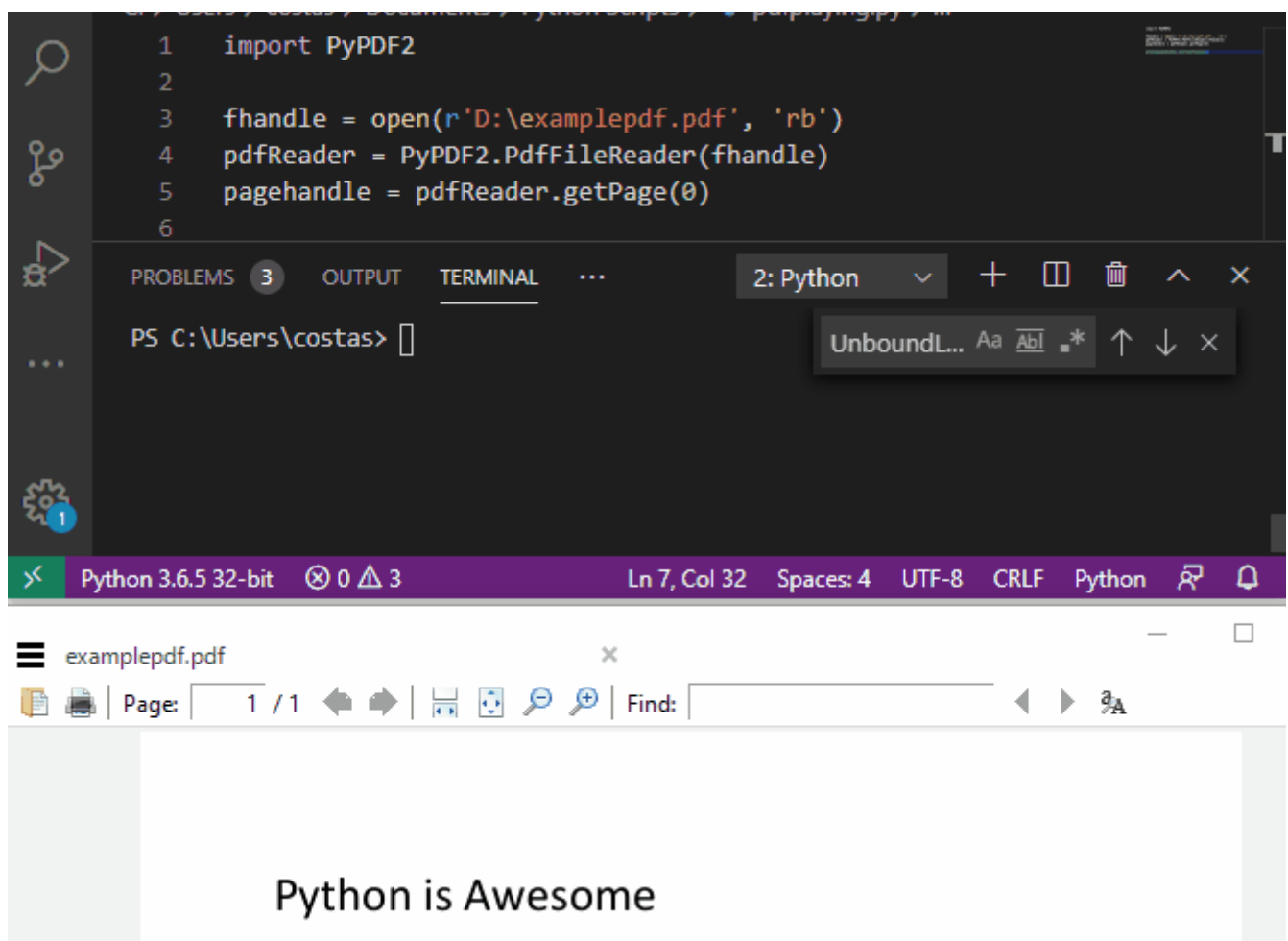
. . .

PyPDF2

Rating: 3/5

The good news with PyPDF2 was that it was a breeze to install. The documentation is somewhat lacking easy examples to follow, but pay close enough attention, and you can figure it out eventually.

The bad news, however, is that the results were not great.



As you can see, it identified the right text, but for some reason, it broke it up into multiple lines.

The code:

```
import PyPDF2

fhandle = open(r'D:\examplepdf.pdf', 'rb')

pdfReader = PyPDF2.PdfFileReader(fhandle)

pagehandle = pdfReader.getPage(0)

print(pagehandle.extractText())
```

. . .

Texttract

Rating: 0/5

Off to a promising start with the number of people raving about this library. The documentation is also good.

Unfortunately, the latest version has a bug which throws an error every time you try to extract text from a PDF. Following the bug through the library's dev forum, there may be a fix in the works. Fingers crossed.

. . .

Apache Tika

Rating: 2/5

Apache Tika has a python library which apparently lets you extract text from PDFs. Installing the Python library is simple enough, but it will not work unless you have JAVA installed.

At least that is the theory. I did not want to install JAVA; hence I remained at:

"RuntimeError: Unable to start Tika server." error.

According to this medium blog (no affiliation), however, once you get it working, it is terrific. So, let's go with 2/5 rating.

The code would apparently look something like:

```
from tika import parser

file = r'D:\examplepdf.pdf'

file_data = parser.from_file(file)

text = file_data['content']

print(text)
```

. . .

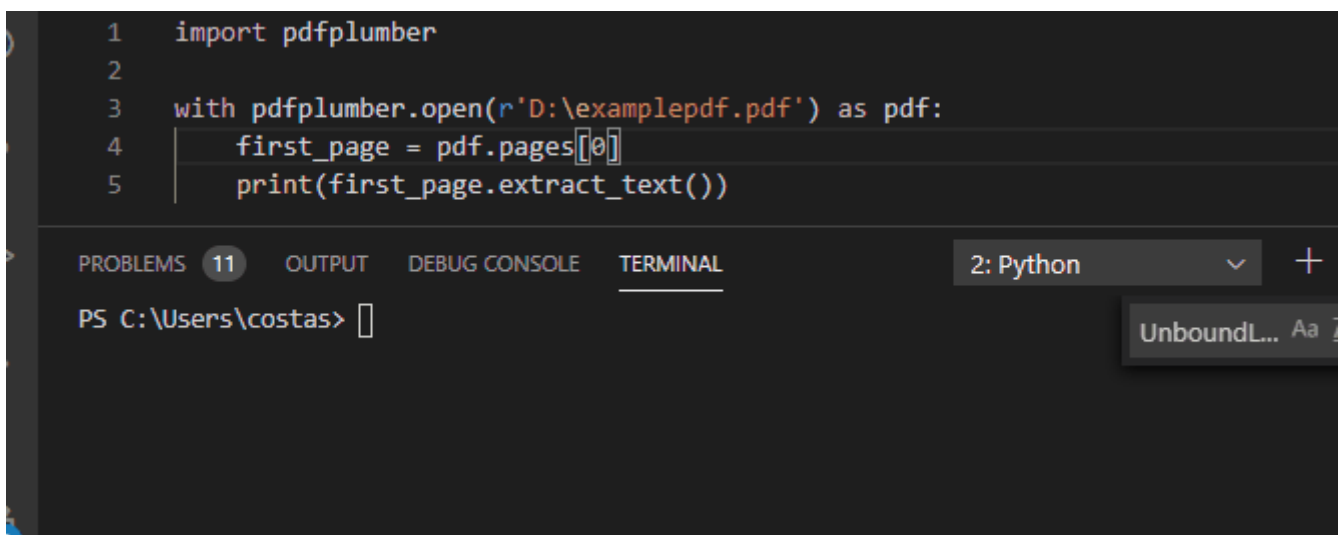
pdfPlumber

Rating: 5/5

Right when I started losing faith in the existence of a simple to use python library for mining text out of pdfs, across comes pdfPlumber.

The documentation is not too bad; within minutes, the whole thing gets going. The results are as good as they can be.

Worth noting, however, that the library does specifically say that it works best on machine-generated PDFs rather than scanned documents; which is what I used.



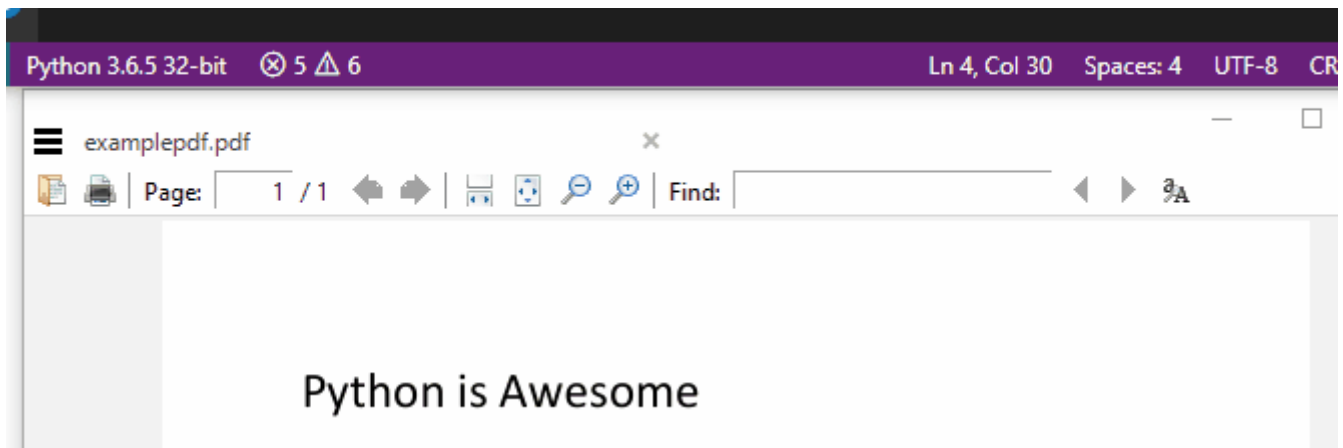
```
1 import pdfplumber
2
3 with pdfplumber.open(r'D:\examplepdf.pdf') as pdf:
4     first_page = pdf.pages[0]
5     print(first_page.extract_text())
```

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\costas>

2: Python

UnboundL... Aa



The code:

```
import pdfplumber

with pdfplumber.open(r'D:\examplepdf.pdf') as pdf:
    first_page = pdf.pages[0]
    print(first_page.extract_text())
```

. . .

pdfMiner3

Rating: 4/5

I will be honest; in a typical pythonic way, I glanced at the documentation (twice!) and failed to understand how I was meant to run this package; this includes pdfMiner (not version 3 that I am reviewing here, as well). I even installed it and tried a few things with no success.

Alas, to my rescue comes a kind stranger in StackOverflow. Once you go through the example provided, it is actually easy to follow. Oh, and the results are as good as you would expect:

```
1  from pdfminer3.layout import LAParams, LTTextBox
2  from pdfminer3.pdfpage import PDFPage
3  from pdfminer3.pdfinterp import PDFResourceManager
4  from pdfminer3.pdfinterp import PDFPageInterpreter
5  from pdfminer3.converter import PDFPageAggregator
6  from pdfminer3.converter import TextConverter
7  import io
8
9  resource_manager = PDFResourceManager()
```

```
10 fake_file_handle = io.StringIO()
11 converter = TextConverter(resource_manager, fake_file_handle, laparams=LAParams)
12 page_interpreter = PDFPageInterpreter(resource_manager, converter)
13
14 with open(r'D:\examplepdf.pdf', 'rb') as fh:
15
16     for page in PDFPage.get_pages(fh,
17                                   caching=True,
18                                   check_extractable=True):
19         page_interpreter.process_page(page)
20
21     text = fake_file_handle.getvalue()
22
23 # close open handles
24 converter.close()
25 fake_file_handle.close()
26
27 print(text)
```

PROBLEMS 17 OUTPUT DEBUG CONSOLE TERMINAL

2: Python

PS C:\Users\costas> []

The code can be found in the linked StackOverflow post.

. . .

PDF -> JPEG -> Text

Another way that this problem could be addressed is by transforming the PDF file into an image. This could be done either programmatically or by taking a screenshot of each page.

Once you have the image files, you can use the tesseract library to extract the text out of them:

How to Extract Text from Images with Python

Learn to extract text from images in 3 lines of codes

towardsdatascience.com

• • •

Before you go, if you liked this article, you may also like:

Learning to Use Progress Bars in Python

Introduction to 4 different libraries (Command Line & UI)

towardsdatascience.com

Building a Python UI for Comparing Data

How to quickly enable your non-technical team to compare data

towardsdatascience.com

[Programming](#)

[Data Science](#)

[Startup](#)

[Python](#)

[Pdf](#)

[About](#) [Help](#) [Legal](#)

Get the Medium app

