



PYTHON
ACADEMY



MANIPULANDO ARQUIVOS PDF COM PYTHON

Nesse ebook você vai aprender a manipular arquivos PDF com Python e PyPDF2

[PYTHONACADEMY.COM.BR](https://pythonacademy.com.br)

Gere ebooks como este com



em <https://ebookr.ai>

Crie ebooks profissionais incríveis em minutos com IA



Chega de formatar texto no Google Docs, Word ou ferramentas que só te fazem perder tempo...

E deixe que nossa IA faça o trabalho pesado!



Capas gerados por IA



Infográficos feitos por IA



Edite em Markdown em Tempo Real



Adicione Banners Promocionais

TESTE AGORA

PRIMEIRO CAPÍTULO 100% GRÁTIS

Salve salve Pythonista!

Neste artigo, vamos explorar como manipular arquivos PDF usando a linguagem de programação Python.

Veremos como criar arquivos PDF, ler texto de arquivos existentes, combinar vários arquivos PDF em um único e até mesmo dividir um arquivo PDF em páginas separadas.

Vamos nessa!

Os arquivos PDF são amplamente utilizados para compartilhar documentos eletrônicos, como relatórios, e-books, manuais, entre outros.

Saber manipular esses arquivos usando programação pode trazer muitos benefícios, como automatizar tarefas relacionadas a documentos, extrair informações de arquivos, criar relatórios personalizados e até mesmo modificar documentos existentes para atender a requisitos específicos.

Felizmente, a biblioteca `PyPDF2` oferece uma maneira fácil de trabalhar com arquivos PDF em Python.

Vamos explorar os principais recursos desta biblioteca nesse artigo!

Criando um arquivo PDF

Vamos começar criando um arquivo PDF do zero!

Para fazer isso, precisamos instalar a biblioteca PyPDF2 usando o gerenciador de pacotes pip.

Primeiro, crie e ative um Ambiente Virtual para esse projeto.

E se ainda não sabe o que é um Ambiente Virtual, pare tudo e [leia esse artigo onde vou te mostrar como criar e utilizar Ambientes Virtuais com Virtualenv](#)

Com seu ambiente virtual ativo, abra o terminal e execute o seguinte comando:

```
pip install PyPDF2
```

Agora, vamos ao código!

Primeiro, importaremos a classe `PdfWriter` da biblioteca PyPDF2 e criaremos um objeto `PdfWriter`.

Em seguida, adicionaremos uma página ao arquivo PDF e salvaremos o arquivo final no disco.

```
from PyPDF2 import PdfWriter

# Criando um objeto PdfWriter
pdf_writer = PdfWriter()

# Adicionando uma página ao arquivo PDF
pdf_writer.add_page()

# Salvando o arquivo PDF no disco
with open("meu_arquivo.pdf", "wb") as output_pdf:
    pdf_writer.write(output_pdf)
```

No código acima, criamos um objeto `PdfWriter` usando `PdfWriter()` e, em seguida, adicionamos uma página ao arquivo PDF usando o método `add_page()`.

Por fim, salvamos o arquivo final usando `write(output_pdf)` e especificando o nome do arquivo como `"meu_arquivo.pdf"`.

Ao executar esse código, um arquivo PDF vazio será criado e salvo no diretório atual.

Lendo texto de um arquivo PDF

Agora que sabemos como criar um arquivo PDF, vamos aprender como ler o texto de um arquivo PDF existente.

Para isso, podemos usar a classe `PdfFileReader` da biblioteca PyPDF2.

Vamos abrir um arquivo PDF existente, extrair o texto dele e exibi-lo no console.

```
from PyPDF2 import PdfFileReader

# Abrindo um arquivo PDF existente
with open("meu_arquivo.pdf", "rb") as input_pdf:
    # Criando um objeto PdfFileReader
    pdf_reader = PdfFileReader(input_pdf)

    # Obtendo o número de páginas do arquivo PDF
    num_pages = pdf_reader.numPages

    # Lendo o texto de cada página
    for page_number in range(num_pages):
        page = pdf_reader.getPage(page_number)
        text = page.extractText()
        print("Texto da página", page_number + 1, ":", text)
```

No código acima, abrimos um arquivo PDF existente usando `open()` e o modo "rb" (leitura binária).

Em seguida, criamos um objeto `PdfFileReader` usando `PdfFileReader(input_pdf)`.

Usamos o método `numPages` para obter o número total de páginas do arquivo PDF.

Depois, usamos um loop `for` para iterar sobre cada página do arquivo PDF.

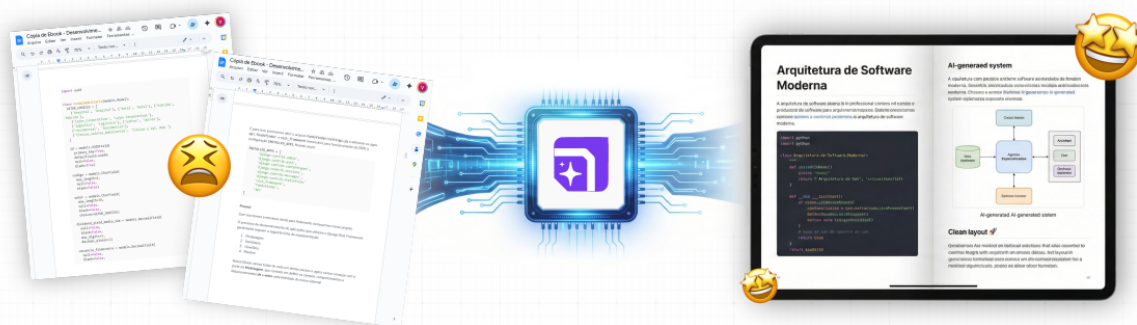
Usamos `getPage(page_number)` para obter uma página específica e `extractText()` para extrair o texto dessa página.

Finalmente, imprimimos o texto resultante no console.



Estou construindo o [Ebookr.ai](https://ebookr.ai), uma plataforma onde você cria ebooks profissionais com IA sobre qualquer assunto — do zero ao PDF pronto, com capas e infográficos gerados automaticamente. Dá uma olhada!


Crie Ebooks profissionais incríveis em minutos com IA




Chega de formatar texto no Google Docs, Word ou ferramentas que só te fazem perder tempo...

... e deixe que nossa IA faça o trabalho pesado!

TESTE AGORA! PRIMEIRO CAPÍTULO 100% GRÁTIS 

 Capas gerados por IA

 Adicione Banners Promocionais

 Edite em Markdown em Tempo Real

 Infográficos feitos por IA

Combinando arquivos PDF

Às vezes, é necessário combinar vários arquivos PDF em um único arquivo.

Felizmente, a biblioteca PyPDF2 oferece suporte a essa funcionalidade.

Vamos criar um exemplo simples para combinar dois arquivos PDF em um único arquivo.

```
from PyPDF2 import PdfFileMerger

# Criando um objeto PdfFileMerger
pdf_merger = PdfFileMerger()

# Adicionando os arquivos PDF ao objeto PdfFileMerger
pdf_merger.append("arquivo1.pdf")
pdf_merger.append("arquivo2.pdf")

# Salvando o arquivo combinado
with open("arquivo_combinado.pdf", "wb") as output_pdf:
    pdf_merger.write(output_pdf)
```

No código acima, criamos um objeto `PdfFileMerger` e adicionamos dois arquivos PDF usando o método `append()`.

Em seguida, salvamos o arquivo combinado usando `write(output_pdf)`.

Dividindo um arquivo PDF

Às vezes, temos um arquivo PDF grande e precisamos dividi-lo em várias páginas separadas. Vamos ver como fazer isso usando a biblioteca PyPDF2.


```

from PyPDF2 import PdfFileReader, PdfFileWriter

# Abrindo o arquivo PDF original
with open("arquivo_grande.pdf", "rb") as input_pdf:
    # Criando um objeto PdfFileReader
    pdf_reader = PdfFileReader(input_pdf)

    # Obtendo o número de páginas do arquivo PDF
    num_pages = pdf_reader.numPages

    # Dividindo o arquivo em várias páginas separadas
    for page_number in range(num_pages):
        # Criando um objeto PdfFileWriter
        pdf_writer = PdfFileWriter()

        # Copiando a página atual para o novo arquivo
        page = pdf_reader.getPage(page_number)
        pdf_writer.addPage(page)

        # Salvando a página separada
        with open(f"pagina_{page_number + 1}.pdf", "wb") as output_pdf:
            pdf_writer.write(output_pdf)

```

No código acima, abrimos o arquivo PDF original usando `open()` e o modo “rb”.

Criamos um objeto `PdfFileReader` a partir desse arquivo e usamos `numPages` para obter o número total de páginas.

Usamos um loop `for` para iterar sobre cada página do arquivo PDF original.

Em cada iteração, criamos um novo objeto `PdfFileWriter` e copiamos a página atual para esse novo arquivo usando `addPage()`.

Em seguida, salvamos a página separada no disco usando `write(output_pdf)`.

Conclusão

Neste artigo, vimos como manipular arquivos PDF usando Python.

Aprendemos como criar arquivos PDF do zero, ler texto de arquivos existentes, combinar vários arquivos PDF em um único e dividir um arquivo PDF em páginas separadas.

Essas habilidades podem ser úteis em uma variedade de cenários, desde automatizar tarefas relacionadas a documentos até extrair informações úteis de arquivos PDF existentes.

A biblioteca PyPDF2 é uma ferramenta poderosa e fácil de usar para o trabalho com arquivos PDF em Python.

Recomendo que você explore mais recursos dessa biblioteca e experimente diferentes manipulações de arquivo PDF por conta própria.

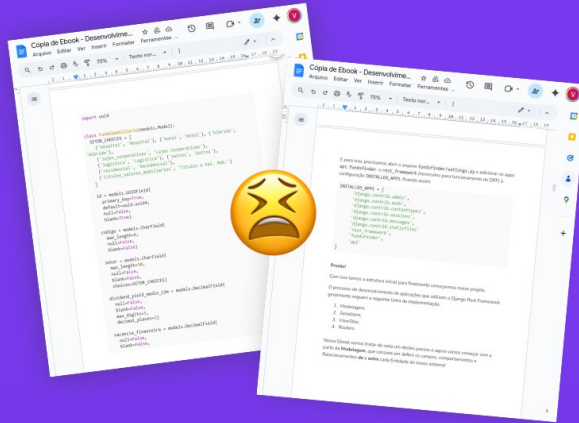
Até a próxima!

Não se esqueça de conferir!

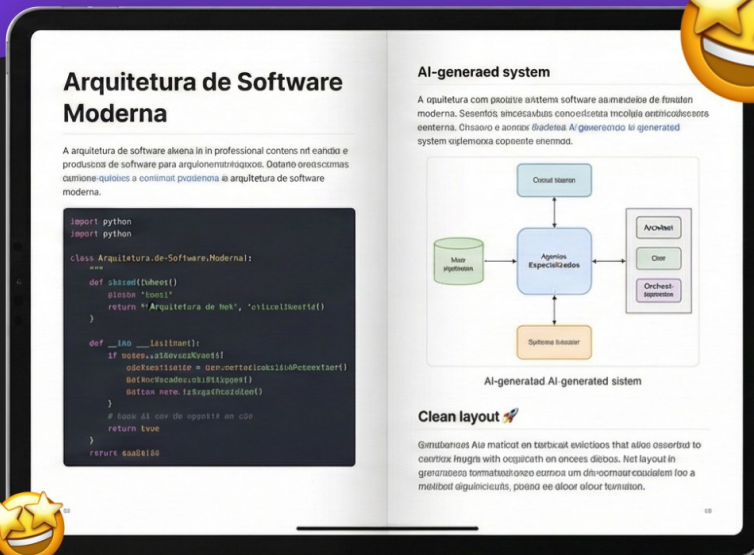


Ebookr

Crie Ebooks profissionais em minutos com IA



Chega de formatar código no Google Docs ou Word



Capas gerados por IA



Infográficos feitos por IA



Adicione Banners Promocionais

Deixe que nossa IA faça o trabalho pesado



Edite em Markdown em Tempo Real

TESTE AGORA



PRIMEIRO CAPÍTULO 100% GRÁTIS