## Remove images

In [ ]:
```
from PyPDF2 import PdfReader, PdfWriter

reader = PdfReader("example.pdf")
writer = PdfWriter()

for page in reader.pages:
    writer.add_page(page)

writer.remove_images()

with open("example3.pdf", "wb") as f:
    writer.write(f)
```

## Removing duplication

In [ ]:
```
from PyPDF2 import PdfReader, PdfWriter

reader = PdfReader("example.pdf")
writer = PdfWriter()

for page in reader.pages:
    writer.add_page(page)

writer.add_metadata(reader.metadata)

with open("example4.pdf", "wb") as fp:
    writer.write(fp)
```

## Lossless Compression

In [ ]:
```
from PyPDF2 import PdfReader, PdfWriter

reader = PdfReader("example.pdf")
writer = PdfWriter()

for page in reader.pages:
    page.compress_content_streams()  # This is CPU intensive!
    writer.add_page(page)

with open("example5.pdf", "wb") as f:
    writer.write(f)
```

## Extract Images

In [ ]:
```
from PyPDF2 import PdfReader

reader = PdfReader("example.pdf")

page = reader.pages[0]
count = 0
```

```
for image_file_object in page.images:
    with open("id" + str(count) + "-" + image_file_object.name, "wb") as fp:
        fp.write(image_file_object.data)
        count += 1
```

## Rotate PDF

In [ ]:

```
# importing the required modules
import PyPDF2

def PDFrotate(origFileName, newFileName, rotation):
    # creating a pdf File object of original pdf
    pdfFileObj = open(origFileName, 'rb')
    # creating a pdf Reader object
    pdfReader = PyPDF2.PdfReader(pdfFileObj)
    # creating a pdf writer object for new pdf
    pdfWriter = PyPDF2.PdfWriter()
    # rotating each page
    for page in range(len(pdfReader.pages)):
        # creating rotated page object
        pageObj = pdfReader.pages[page]
        pageObj.rotate(rotation)
        # adding rotated page object to pdf writer
    pdfWriter.add_page(pageObj)
    # new pdf file object
    newFile = open(newFileName, 'wb')
    # writing rotated pages to new file
    pdfWriter.write(newFile)
    # closing the original pdf file object
    pdfFileObj.close()
    # closing the new pdf file object
    newFile.close()

def main():
    # original pdf file name
    origFileName = 'example.pdf'
    # new pdf file name
    newFileName = 'rotated_example.pdf'
    # rotation angle
    rotation = 90
    # calling the PDFrotate function
    PDFrotate(origFileName, newFileName, rotation)

if __name__ == "__main__":
    # calling the main function
    main()
    print("rotated pdf by 90 degree")
```

## Merger PDF

In [ ]:

```
import PyPDF2

def PDFmerge(pdfs, output):
    # creating pdf file merger object
    pdfMerger = PyPDF2.PdfMerger()

    # appending pdfs one by one
```

```
    for pdf in pdfs:
        pdfMerger.append(pdf)

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

def main():
    # pdf files to merge
    pdfs = ['example1.pdf', 'example2.pdf']

    # output pdf file name
    output = 'example12.pdf'

    # calling pdf merge function
    PDFmerge(pdfs=pdfs, output=output)

if __name__ == "__main__":
    # calling the main function
    main()
    print("merge pdf task completed")
```

## Read PDF

In [ ]:
```
import PyPDF2

pdfFileObj = open('example.pdf', 'rb')

# creating a pdf reader object
pdfReader = PyPDF2.PdfReader(pdfFileObj)
print(len(pdfReader.pages))

pageObj = pdfReader.pages[0]
print(pageObj.extract_text())

pdfFileObj.close()
```

## Encrypt PDF

In [ ]:
```
from PyPDF2 import PdfReader, PdfWriter
password = "daio_2023"

reader = PdfReader("example.pdf")
writer = PdfWriter()

# Add all pages to the writer
for page in reader.pages:
    writer.add_page(page)

# Add a password to the new PDF
writer.encrypt(password)

# Save the new PDF to a file
with open("encrypted-example.pdf", "wb") as f:
    writer.write(f)
```

## Decrypt PDF

In [ ]:
```
from PyPDF2 import PdfReader, PdfWriter
password = "daio_2023"

reader = PdfReader("encrypted-example.pdf")
writer = PdfWriter()

if reader.is_encrypted:
    reader.decrypt(password)

# Add all pages to the writer
for page in reader.pages:
    writer.add_page(page)

# Save the new PDF to a file
with open("decrypted-example.pdf", "wb") as f:
    writer.write(f)
```

## Reading PDFs directly from Google cloud services

In [ ]:
```
import gcsfs
from pypdf import PdfReader

gcs_file_system = gcsfs.GCSFileSystem(project="PROJECT_ID")
gcs_pdf_path = "gs://bucket_name/object.pdf"

f_object = gcs_file_system.open(gcs_pdf_path, "rb")

# Open our PDF file with the PdfReader
reader = PdfReader(f_object)

# Get number of pages
num = len(reader.pages)

f_object.close()
```

## Reading PDFs directly from AWS cloud services

In [ ]:
```
from io import BytesIO

import boto3
from PyPDF2 import PdfReader

s3 = boto3.client("s3")
obj = s3.get_object(Body=csv_buffer.getvalue(), Bucket="my-bucket", Key="my/doc.pdf")
reader = PdfReader(BytesIO(obj["Body"].read()))
```

## Writing a PDF directly to AWS S3

In [ ]:
```
from io import BytesIO

import boto3
from PyPDF2 import PdfReader, PdfWriter
```

```
reader = PdfReader(BytesIO(raw_bytes_data))
writer = PdfWriter()

# Add all pages to the writer
for page in reader.pages:
    writer.add_page(page)

# Add a password to the new PDF
writer.encrypt("my-secret-password")

# Save the new PDF to a file
with BytesIO() as bytes_stream:
    writer.write(bytes_stream)
    bytes_stream.seek(0)
    s3 = boto3.client("s3")
    s3.write_get_object_response(
        Body=bytes_stream, RequestRoute=request_route, RequestToken=request_token
    )
```

## Streaming Data with PyPDF2

In [ ]:
```
from io import BytesIO

# Prepare example
with open("example.pdf", "rb") as fh:
    bytes_stream = BytesIO(fh.read())

# Read from bytes_stream
reader = PdfReader(bytes_stream)

# Write to bytes_stream
writer = PdfWriter()
with BytesIO() as bytes_stream:
    writer.write(bytes_stream)
```

## Interactions with PDF Forms

### Reading form fields

In [ ]:
```
from PyPDF2 import PdfReader

reader = PdfReader("form.pdf")
fields = reader.get_form_text_fields()
fields == {"key": "value", "key2": "value2"}
```

### Filling out forms

In [ ]:
```
from PyPDF2 import PdfReader, PdfWriter

reader = PdfReader("form.pdf")
writer = PdfWriter()

page = reader.pages[0]
```

```
writer.add_page(page)

writer.update_page_form_field_values(
    writer.pages[0], {"fieldname": "some filled in text"}
)

# write "output" to PyPDF2-output.pdf
with open("filled-out.pdf", "wb") as output_stream:
    writer.write(output_stream)
```

# Adding PDF Annotations

## Attachments

In [ ]:
```
from PyPDF2 import PdfWriter

writer = PdfWriter()
writer.add_blank_page(width=200, height=200)

data = b"any bytes - typically read from a file"
writer.add_attachment("smile.png", data)

with open("output.pdf", "wb") as output_stream:
    writer.write(output_stream)
```

## Free Text

In [ ]:
```
from PyPDF2 import PdfReader, PdfWriter
from PyPDF2.generic import AnnotationBuilder

# Fill the writer with the pages you want
pdf_path = os.path.join(RESOURCE_ROOT, "crazyones.pdf")
reader = PdfReader(pdf_path)
page = reader.pages[0]
writer = PdfWriter()
writer.add_page(page)

# Create the annotation and add it
annotation = AnnotationBuilder.free_text(
    "Hello World\nThis is the second line!",
    rect=(50, 550, 200, 650),
    font="Arial",
    bold=True,
    italic=True,
    font_size="20pt",
    font_color="00ff00",
    border_color="0000ff",
    background_color="cdcdcd",
)
writer.add_annotation(page_number=0, annotation=annotation)

# Write the annotated file to disk
with open("annotated-pdf.pdf", "wb") as fp:
    writer.write(fp)
```

## Line

In [ ]:
```
pdf_path = os.path.join(RESOURCE_ROOT, "crazyones.pdf")
reader = PdfReader(pdf_path)
page = reader.pages[0]
writer = PdfWriter()
writer.add_page(page)

# Add the line
annotation = AnnotationBuilder.line(
    text="Hello World\nLine2",
    rect=(50, 550, 200, 650),
    p1=(50, 550),
    p2=(200, 650),
)
writer.add_annotation(page_number=0, annotation=annotation)

# Write the annotated file to disk
with open("annotated-pdf.pdf", "wb") as fp:
    writer.write(fp)
```

## Rectangle

In [ ]:
```
pdf_path = os.path.join(RESOURCE_ROOT, "crazyones.pdf")
reader = PdfReader(pdf_path)
page = reader.pages[0]
writer = PdfWriter()
writer.add_page(page)

# Add the line
annotation = AnnotationBuilder.rectangle(
    rect=(50, 550, 200, 650),
)
writer.add_annotation(page_number=0, annotation=annotation)

# Write the annotated file to disk
with open("annotated-pdf.pdf", "wb") as fp:
    writer.write(fp)
```

## Link

In [ ]:
```
pdf_path = os.path.join(RESOURCE_ROOT, "crazyones.pdf")
reader = PdfReader(pdf_path)
page = reader.pages[0]
writer = PdfWriter()
writer.add_page(page)

# Add the line
annotation = AnnotationBuilder.link(
    rect=(50, 550, 200, 650),
    url="https://martin-thoma.com/",
)
writer.add_annotation(page_number=0, annotation=annotation)

# Write the annotated file to disk
```

```
with open("annotated-pdf.pdf", "wb") as fp:
    writer.write(fp)
```

### Link #2

In [ ]:
```
pdf_path = os.path.join(RESOURCE_ROOT, "crazyones.pdf")
reader = PdfReader(pdf_path)
page = reader.pages[0]
writer = PdfWriter()
writer.add_page(page)

# Add the line
annotation = AnnotationBuilder.link(
    rect=(50, 550, 200, 650), target_page_index=3, fit="/FitH", fit_args=(123,)
)
writer.add_annotation(page_number=0, annotation=annotation)

# Write the annotated file to disk
with open("annotated-pdf.pdf", "wb") as fp:
    writer.write(fp)
```

## Reading PDF Annotations

### General Annotations

In [ ]:
```
from PyPDF2 import PdfReader

reader = PdfReader("commented.pdf")

for page in reader.pages:
    if "/Annots" in page:
        for annot in page["/Annots"]:
            obj = annot.get_object()
            annotation = {"subtype": obj["/Subtype"], "location": obj["/Rect"]}
            print(annotation)
```

### Text

In [ ]:
```
from PyPDF2 import PdfReader

reader = PdfReader("example.pdf")

for page in reader.pages:
    if "/Annots" in page:
        for annot in page["/Annots"]:
            subtype = annot.get_object()["/Subtype"]
            if subtype == "/Text":
                print(annot.get_object()["/Contents"])
```

### Highlights

In [ ]:
```
from PyPDF2 import PdfReader

reader = PdfReader("commented.pdf")
```

```
for page in reader.pages:
    if "/Annots" in page:
        for annot in page["/Annots"]:
            subtype = annot.get_object()["/Subtype"]
            if subtype == "/Highlight":
                coords = annot.get_object()["/QuadPoints"]
                x1, y1, x2, y2, x3, y3, x4, y4 = coords
```

### Attachments

In [ ]:
```
from PyPDF2 import PdfReader

reader = PdfReader("example.pdf")

attachments = {}
for page in reader.pages:
    if "/Annots" in page:
        for annotation in page["/Annots"]:
            subtype = annot.get_object()["/Subtype"]
            if subtype == "/FileAttachment":
                fileobj = annotobj["/FS"]
                attachments[fileobj["/F"]] = fileobj["/EF"]["/F"].get_data()
```

## Adding a Stamp/Watermark to a PDF

### Stamp (Overlay)

In [ ]:
```
from pathlib import Path
from typing import Union, Literal, List

from PyPDF2 import PdfWriter, PdfReader


def stamp(
    content_pdf: Path,
    stamp_pdf: Path,
    pdf_result: Path,
    page_indices: Union[Literal["ALL"], List[int]] = "ALL",
):
    reader = PdfReader(stamp_pdf)
    image_page = reader.pages[0]

    writer = PdfWriter()

    reader = PdfReader(content_pdf)
    if page_indices == "ALL":
        page_indices = list(range(0, len(reader.pages)))
    for index in page_indices:
        content_page = reader.pages[index]
        mediabox = content_page.mediabox
        content_page.merge_page(image_page)
        content_page.mediabox = mediabox
        writer.add_page(content_page)

    with open(pdf_result, "wb") as fp:
        writer.write(fp)
```

### Watermark (Underlay)

In [ ]:
```
from pathlib import Path
from typing import Union, Literal, List

from PyPDF2 import PdfWriter, PdfReader


def watermark(
    content_pdf: Path,
    stamp_pdf: Path,
    pdf_result: Path,
    page_indices: Union[Literal["ALL"], List[int]] = "ALL",
):
    reader = PdfReader(content_pdf)
    if page_indices == "ALL":
        page_indices = list(range(0, len(reader.pages)))

    writer = PdfWriter()
    for index in page_indices:
        content_page = reader.pages[index]
        mediabox = content_page.mediabox

        # You need to load it again, as the last time it was overwritten
        reader_stamp = PdfReader(stamp_pdf)
        image_page = reader_stamp.pages[0]

        image_page.merge_page(content_page)
        image_page.mediabox = mediabox
        writer.add_page(image_page)

    with open(pdf_result, "wb") as fp:
        writer.write(fp)
```

## Cropping and Transforming PDFs

In [ ]:
```
from PyPDF2 import PdfWriter, PdfReader

reader = PdfReader("example.pdf")
writer = PdfWriter()

# add page 1 from reader to output document, unchanged:
writer.add_page(reader.pages[0])

# add page 2 from reader, but rotated clockwise 90 degrees:
writer.add_page(reader.pages[1].rotate(90))

# add page 3 from reader, but crop it to half size:
page3 = reader.pages[2]
page3.mediabox.upper_right = (
    page3.mediabox.right / 2,
    page3.mediabox.top / 2,
)
writer.add_page(page3)

# add some Javascript to launch the print window on opening this PDF.
# the password dialog may prevent the print dialog from being shown,
```

```
# comment the the encription lines, if that's the case, to try this out:
writer.add_js("this.print({bUI:true,bSilent:false,bShrinkToFit:true});")

# write to document-output.pdf
with open("PyPDF2-output.pdf", "wb") as fp:
    writer.write(fp)
```

## Page rotation

In [ ]:
```
from PyPDF2 import PdfWriter, PdfReader

reader = PdfReader("input.pdf")
writer = PdfWriter()


writer.add_page(reader.pages[0])
writer.pages[0].rotate(90)

with open("output.pdf", "wb") as fp:
    writer.write(fp)
```

## Plain Merge

In [ ]:
```
from PyPDF2 import PdfReader, PdfWriter, Transformation

# Get the data
reader_base = PdfReader("labeled-edges-center-image.pdf")
page_base = reader_base.pages[0]

reader = PdfReader("box.pdf")
page_box = reader.pages[0]

page_base.merge_page(page_box)

# Write the result back
writer = PdfWriter()
writer.add_page(page_base)
with open("merged-foo.pdf", "wb") as fp:
    writer.write(fp)
```

## Merge with Rotation

In [ ]:
```
from PyPDF2 import PdfReader, PdfWriter, Transformation

# Get the data
reader_base = PdfReader("labeled-edges-center-image.pdf")
page_base = reader_base.pages[0]

reader = PdfReader("box.pdf")
page_box = reader.pages[0]

# Apply the transformation
transformation = Transformation().rotate(45)
page_box.add_transformation(transformation)
page_base.merge_page(page_box)

# Write the result back
```

```
writer = PdfWriter()
writer.add_page(page_base)
with open("merged-foo.pdf", "wb") as fp:
    writer.write(fp)
```

## Scaling a Page (the Canvas)

In [ ]:
```
from PyPDF2 import PdfReader, PdfWriter

# Read the input
reader = PdfReader("resources/side-by-side-subfig.pdf")
page = reader.pages[0]

# Scale
page.scale_by(0.5)

# Write the result to a file
writer = PdfWriter()
writer.add_page(page)
writer.write("out.pdf")
```

from PyPDF2.generic import RectangleObject mb = page.mediabox page.mediabox = RectangleObject((mb.left, mb.bottom, mb.right, mb.top)) page.cropbox = RectangleObject((mb.left, mb.bottom, mb.right, mb.top)) page.trimbox = RectangleObject((mb.left, mb.bottom, mb.right, mb.top)) page.bleedbox = RectangleObject((mb.left, mb.bottom, mb.right, mb.top)) page.artbox = RectangleObject((mb.left, mb.bottom, mb.right, mb.top))

## Scaling the content

In [ ]:
```
from PyPDF2 import PdfReader, PdfWriter, Transformation

# Read the input
reader = PdfReader("resources/side-by-side-subfig.pdf")
page = reader.pages[0]

# Scale
op = Transformation().scale(sx=0.7, sy=0.7)
page.add_transformation(op)

# Write the result to a file
writer = PdfWriter()
writer.add_page(page)
writer.write("out-pg-transform.pdf")
```

## Merging PDF files with more options

In [ ]:
```
from PyPDF2 import PdfWriter

merger = PdfWriter()

input1 = open("document1.pdf", "rb")
input2 = open("document2.pdf", "rb")
input3 = open("document3.pdf", "rb")

# add the first 3 pages of input1 document to output
merger.append(fileobj=input1, pages=(0, 3))
```

```
# insert the first page of input2 into the output beginning after the second page
merger.merge(position=2, fileobj=input2, pages=(0, 1))

# append entire input3 document to the end of the output document
merger.append(input3)

# Write to an output PDF document
output = open("document-output.pdf", "wb")
merger.write(output)

# Close File Descriptors
merger.close()
output.close()
```

## Extract Text from a PDF

In [ ]:
```
from PyPDF2 import PdfReader

reader = PdfReader("example.pdf")
page = reader.pages[0]
print(page.extract_text())
```

# extract only text oriented up print(page.extract_text(0)) # extract text oriented up and turned left print(page.extract_text((0, 90)))

### Ignore header and footer

In [ ]:
```
from PyPDF2 import PdfReader

reader = PdfReader("GeoBase_NHNC1_Data_Model_UML_EN.pdf")
page = reader.pages[3]

parts = []


def visitor_body(text, cm, tm, fontDict, fontSize):
    y = tm[5]
    if y > 50 and y < 720:
        parts.append(text)


page.extract_text(visitor_text=visitor_body)
text_body = "".join(parts)

print(text_body)
```

### Extract rectangles and texts into a SVG-file

In [ ]:
```
from PyPDF2 import PdfReader
import svgwrite

reader = PdfReader("GeoBase_NHNC1_Data_Model_UML_EN.pdf")
page = reader.pages[2]

dwg = svgwrite.Drawing("GeoBase_test.svg", profile="tiny")
```

```
def visitor_svg_rect(op, args, cm, tm):
    if op == b"re":
        (x, y, w, h) = (args[i].as_numeric() for i in range(4))
        dwg.add(dwg.rect((x, y), (w, h), stroke="red", fill_opacity=0.05))


def visitor_svg_text(text, cm, tm, fontDict, fontSize):
    (x, y) = (tm[4], tm[5])
    dwg.add(dwg.text(text, insert=(x, y), fill="blue"))


page.extract_text(
    visitor_operand_before=visitor_svg_rect, visitor_text=visitor_svg_text
)
dwg.save()
```

## Reading metadata

In [ ]:
```
from PyPDF2 import PdfReader

reader = PdfReader("example.pdf")

meta = reader.metadata

print(len(reader.pages))

# All of the following could be None!
print(meta.author)
print(meta.creator)
print(meta.producer)
print(meta.subject)
print(meta.title)
```

## Writing metadata

In [ ]:
```
from PyPDF2 import PdfReader, PdfWriter

reader = PdfReader("example.pdf")
writer = PdfWriter()

# Add all pages to the writer
for page in reader.pages:
    writer.add_page(page)

# Add the metadata
writer.add_metadata(
    {
        "/Author": "Martin",
        "/Producer": "Libre Writer",
    }
)

# Save the new PDF to a file
with open("meta-pdf.pdf", "wb") as f:
    writer.write(f)
```

In [ ]: