PDF'S & WORD DOCUMENTS

Door Joren Beirens & Thomas Van Olmen

K.H.Kempen en Lessius bundelen de krachten en worden *more*.

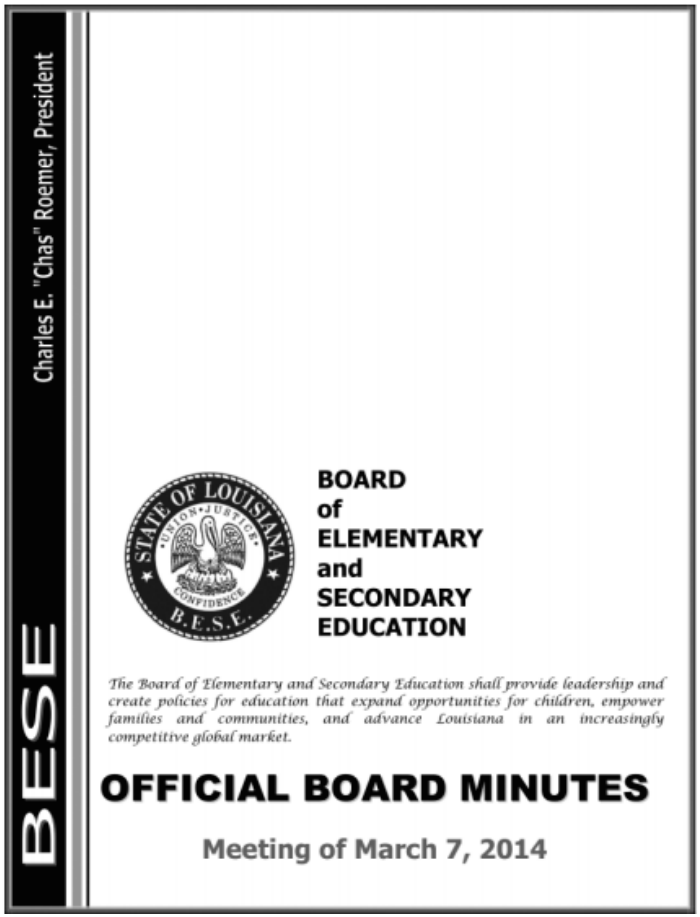
PDF'S LEZEN

- PDF's lezen a.d.h.v. de PyPDF2 module (case sensitive)
- Tekst parsen (pas op! foutenmarge)
- Enkel tekst, geen afbeeldingen/media

COMMANDO'S

- import PyPDF2
- File openen met open('filenaam.pdf', 'rb')
  - rb voor leesmode
  - wb voor schrijfmode
- Aantal pagina's met ".numPages"
- Reader starten met PyPDF2.PdfFileReader()
- Pagina kiezen met ".getPage()"
- ".extractText()" om tekst in strings te krijgen

VOORBEELDPROGRAMMA


BESE
Charles E. "Chas" Roemer, President
BOARD of ELEMENTARY and SECONDARY EDUCATION
The Board of Elementary and Secondary Education shall provide leadership and create policies for education that expand opportunities for children, empower families and communities, and advance Louisiana in an increasingly competitive global market.
OFFICIAL BOARD MINUTES
Meeting of March 7, 2014

VOORBEELDPROGRAMMA

```
>>> import PyPDF2
>>> pdfFileObj = open('meetingminutes.pdf', 'rb')
>>> pdfReader = PyPDF2.PdfFileReader(pdfFileObj)
❶ >>> pdfReader.numPages
19
❷ >>> pageObj = pdfReader.getPage(0)
❸ >>> pageObj.extractText()
'OOFFFFIICCIIAALL  BBOOAARRDD  MMIINNUUTTEESS   Meeting of March 7, 2015
\n     The Board of Elementary and Secondary Education shall provide leadership
and create policies for education that expand opportunities for children,
empower families and communities, and advance Louisiana in an increasingly
competitive global market. BOARD  of ELEMENTARY and  SECONDARY EDUCATION  '
```

ENCRYPTED PDF

- Nakijken of PDF versleuteld is met **“.isEncrypted”**
- Boolean “True” if encrypted
- Decrypt met **.decrypt(‘wachtwoord’)**
  -indien verkeerd: returned 0 en openen faalt
- Wachtwoord toevoegen met **.encrypt()**
  -eerste argument: wachtwoord gebruiker
  -tweede argument: wachtwoord eigenaar

VOORBEELD ENCRYPTIE

```
>>> import PyPDF2
>>> pdfReader = PyPDF2.PdfFileReader(open('encrypted.pdf', 'rb'))
❶ >>> pdfReader.isEncrypted
True

>>> pdfReader.getPage(0)
❷ Traceback (most recent call last):
  File "<pyshell#173>", line 1, in <module>
    pdfReader.getPage()
  --snip--
  File "C:\Python34\lib\site-packages\PyPDF2\pdf.py", line 1173, in getObject
    raise utils.PdfReadError("file has not been decrypted")
PyPDF2.utils.PdfReadError: file has not been decrypted
❸ >>> pdfReader.decrypt('rosebud')
1
>>> pageObj = pdfReader.getPage(0)
```

PDF'S MAKEN & BEWERKEN

1. Bestaande PDF's openen als "lees" object
2. Nieuwe PDF aanmaken als "schrijf" object in schrijfmodus
3. Inhoud van leesobject kopiëren naar schrijfobject
4. D.m.v. Schrijfobject output maken
5. Na bewerken ALLE bestanden opnieuw sluiten

COMMANDO'S

- PyPDF2.PdfFileReader()
- PyPDF2.PdfFileWriter()
- .addPage()
- .write()
- .close()

VOORBEELD PDF'S MAKEN & BEWERKEN

```
>>> import PyPDF2
>>> pdf1File = open('meetingminutes.pdf', 'rb')
>>> pdf2File = open('meetingminutes2.pdf', 'rb')
❶ >>> pdf1Reader = PyPDF2.PdfFileReader(pdf1File)
❷ >>> pdf2Reader = PyPDF2.PdfFileReader(pdf2File)
❸ >>> pdfWriter = PyPDF2.PdfFileWriter()

>>> for pageNum in range(pdf1Reader.numPages):
❹         pageObj = pdf1Reader.getPage(pageNum)
❺         pdfWriter.addPage(pageObj)

>>> for pageNum in range(pdf2Reader.numPages):
❹         pageObj = pdf2Reader.getPage(pageNum)
❺         pdfWriter.addPage(pageObj)

❻ >>> pdfOutputFile = open('combinedminutes.pdf', 'wb')
>>> pdfWriter.write(pdfOutputFile)
>>> pdfOutputFile.close()
>>> pdf1File.close()
>>> pdf2File.close()
```

ROTATING PAGES

- .rotateClockwise()
- .rotateCounterClockwise()
- Enkel de integers 90, 180 of 270

OVERLAYING PAGES

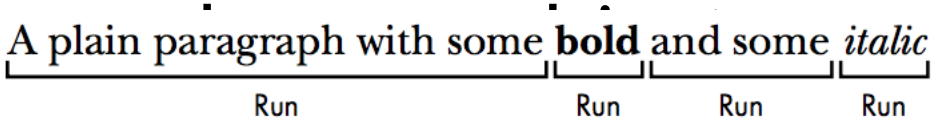
- Om logo's, watermarks toevoegen

1. Een pagina selecteren
2. Met .mergePage("ToeTeVoegenPagina") kan je een andere pagina toevoegen

WORD DOCUMENTS

- **Module python-docx installeren met pip**
- NIET docx! Dit is een andere module

- Paragraph objects bestaan uit één of



- Run objects hebben eigen opmaak

WORD DOCUMENTEN LEZEN

- Bestand openen met .Document('filename')
- .len() voor aantal paragraph objects
- Elk run object heeft attributen
  - Tekst
  - Lettertype
  - …
- Nieuw run object wanneer de textstijl veranderd
- Document > paragraphs > runs

VOORBEELDPROGRAMMA

```
>>> import docx
❶ >>> doc = docx.Document('demo.docx')
❷ >>> len(doc.paragraphs)
7
❸ >>> doc.paragraphs[0].text
'Document Title'
❹ >>> doc.paragraphs[1].text
'A plain paragraph with some bold and some italic'
❺ >>> len(doc.paragraphs[1].runs)
4
❻ >>> doc.paragraphs[1].runs[0].text
'A plain paragraph with some '
❼ >>> doc.paragraphs[1].runs[1].text
'bold'
❽ >>> doc.paragraphs[1].runs[2].text
' and some '
❾ >>> doc.paragraphs[1].runs[3].text
'italic'
```

VOORBEELD FULL TEXT

- Enkel tekst en newlines, geen opmaak
- bijvoorbeeld om te zoeken.

```
#! python3

import docx

def getText(filename):
    doc = docx.Document(filename)
    fullText = []
    for para in doc.paragraphs:
        fullText.append(para.text)
    return '\n'.join(fullText)
```

```
>>> import readDocx
>>> print(readDocx.getText('demo.docx'))
Document Title
A plain paragraph with some bold and some italic
Heading, level 1
Intense quote
first item in unordered list
first item in ordered list
```

STYLING PARAGRAPHS /RUN OBJECTS

- Weergave alle stijlen In Word: Ctrl+Alt+shift+S
- 3 stijltypes:
  - Paragraph style -> paragraphs
  - Character style -> run objects
  - Linked style -> beide

```
'Normal'          'Heading5'        'ListBullet'      'ListParagraph'
'BodyText'        'Heading6'        'ListBullet2'     'MacroText'
'BodyText2'       'Heading7'        'ListBullet3'     'NoSpacing'
'BodyText3'       'Heading8'        'ListContinue'    'Quote'
'Caption'         'Heading9'        'ListContinue2'   'Subtitle'
'Heading1'        'IntenseQuote'    'ListContinue3'   'TOCHeading'
'Heading2'        'List'            'ListNumber'      'Title'
'Heading3'        'List2'           'ListNumber2'
'Heading4'        'List3'           'ListNumber3'
```

RUN OBJECTS STYLING

- !! Opgelet. Als je een runobject een aparte stijl wil geven voeg je *Char* toe aan de stijl.

paragraphObj.style = ‘Quote’

runOBJ.style = ‘QuoteChar’

RUN ATTRIBUTES

- Waardes: True, False, None

Attribute	Description
bold	The text appears in bold.
italic	The text appears in italic.
underline	The text is underlined.
strike	The text appears with strikethrough.
double_strike	The text appears with double strikethrough.
all_caps	The text appears in capital letters.
small_caps	The text appears in capital letters, with lowercase letters two points smaller.
shadow	The text appears with a shadow.
outline	The text appears outlined rather than solid.
rtl	The text is written right-to-left.
imprint	The text appears pressed into the page.
emboss	The text appears raised off the page in relief.

VOORBEELD AANPASSEN STIJL IN TEKST

```
>>> doc = docx.Document('demo.docx')
>>> doc.paragraphs[0].text
'Document Title'
>>> doc.paragraphs[0].style
'Title'
>>> doc.paragraphs[0].style = 'Normal'
>>> doc.paragraphs[1].text
'A plain paragraph with some bold and some italic'
>>> (doc.paragraphs[1].runs[0].text, doc.paragraphs[1].runs[1].text, doc.
paragraphs[1].runs[2].text, doc.paragraphs[1].runs[3].text)
('A plain paragraph with some ', 'bold', ' and some ', 'italic')
>>> doc.paragraphs[1].runs[0].style = 'QuoteChar'
>>> doc.paragraphs[1].runs[1].underline = True
>>> doc.paragraphs[1].runs[3].underline = True
>>> doc.save('restyled.docx')
```

SCHRIJVEN IN WORD DOCUMENTS

- .add_paragraph('textToBeAdded')
  - Paragraaf toevoegen aan het eind van de tekst
- .add_run('nieuwe tekst voor de run')
  - Voegt tekst toe aan bestaande paragraaf
- .add_run('Nieuwerun', 'Title')
  - Tweede optioneel argument dat meegegeven wordt is de stijl
- .save('nieuwBestand.docx')

HEADERS TOEVOEGEN

- .add_heading(‘toe te voegen tekst’ , 0)
  - Voegt een paragraaf toe in de stijl van de header die gedefinieerd werd. Het tweede argument is een integer met een waarde tussen 0 en 4.
  - 0: titelstijl
  - 1: main heading
  - 2-4: kleinere subheadings

BREAKS

- Line break:
  - .add_break() toevoegen aan run argument
- Page break:
  - .add_break(docx.text.WD_BREAK.PAGE)

```
>>> doc = docx.Document()
>>> doc.add_paragraph('This is on the first page!')
<docx.text.Paragraph object at 0x0000000003785518>
>>> doc.paragraphs[0].runs[0].add_break(docx.text.WD_BREAK.PAGE)
>>> doc.add_paragraph('This is on the second page!')
<docx.text.Paragraph object at 0x00000000037855F8>
>>> doc.save('twoPage.docx')
```

AFBEELDINGEN TOEVOEGEN

- .add_picture('targetbestand.png')
- Optioneel height/width in metric of imperial units:

```
>>> doc.add_picture('zophie.png', width=docx.shared.Inches(1),
height=docx.shared.Cm(4))
```

VRAGEN (1/2)

- Er wordt geen string value aan de functie PyPDF2 gegeven, maar wat geef je wel mee?
- In welke mode moeten file objecten geopend zijn voor de PdfFileReader() en PdfFileWriter()?
- In welke PdfFileReader variabele wordt het aantal pagina's opgeslagen?
- Als een PDF geëncrypteerd is met het wachtwoord "swordfish", wat moet je doen vooraleer je pagina-objecten kan verkrijgen?
- Met welke methode kan je een pagina draaien?

VRAGEN (2/2)

- Wat is het verschil tussen een paragraaf-object en een run-object?
- Hoe kan je een lijst met paragraaf-objecten verkrijgen voor een document-object dat in de variabele “doc” is opgeslagen?
- Welk type object heeft “bold”, “italic”, “strike” en “outline” variabelen?
- Hoe kan je een nieuw document-object maken voor een nieuw Word document?