



PDF Days Europe 2022 | Berlin

# Putting a squeeze on your PDF

*Techniques to reduce PDF file size and the impact on document content*

# About me



## Maksim Bezrukov

- Component Engineering Lead at iText Software
- iText core modules contributor
- Former veraPDF developer

[maksim.bezrukov@itextpdf.com](mailto:maksim.bezrukov@itextpdf.com)



iText Software is a global leader in PDF technology and the code behind the iText PDF library has existed for more than 20 years.

Millions of **iText users**, both open source and commercial.

Spanning financial, public, government and healthcare sectors, including many Fortune 500 companies.



ITEXT

# Introduction



- Techniques to reduce the size of already created PDF files
- Focusing on preserving the visual appearance
- PDF/A, PDF/UA, etc. compatibility of the discussed methods is not covered
- Not for long-term preservation or archiving

# Compression



- Streams compression
  - FlateDecode filter
  - LZWDecode filter
- Object Stream
- Cross Reference Stream

# Redundant PDF Structures



- Unused objects
  - No references in other objects
- Unused resources
  - No resource usage in content stream
  - Be aware about resources inheritance in pages tree structure
- Comments
  - Do not remove required comments: file header, end of file marker
- Duplicates
  - Be aware about the object context. Identical objects may have completely different contexts and merging may affect the resultant PDF in an unexpected way. Example: unfilled forms

# Redundant PDF Structures



- Alternate images
  - May affect visual presentation in cases of opening the PDF file in a viewer which may not process some images
- Thumbnails
- Bookmarks
- Merge Incremental Updates
  - Loosing changes history
  - Commonly used for multi-signed documents

# (Not) Redundant PDF Structures



- Tags
  - Lose accessibility from PDF
- Optional content groups (layers)
  - If layers are disabled by default, then they can be removed, can't they?
  - Can be used for localization of PDF
- Annotations
  - Simple Annotations, Forms, Signatures
  - Some annotations do affect the visual presentation. So, consider **flattening** instead of removing



# Flattening



- Forms
- Other annotations
- Several transparent objects overlapping

# Content Stream



## ■ Simple optimizations

- Setting font without showing text
- Subsequent text positioning
- Subsequent color operators
- Etc.

```
/F1 12 Tf
```

```
...  
(no text show)
```

```
...  
/F2 12 Tf
```

```
50 50 Td
```

```
...  
(no text show)
```

```
...  
100 50 Td
```

```
1 1 1 rg  
0 0 0 rg
```

## ■ Invisible or Overlapping content

```
1 0 0 rg  
100 100 50 50 re  
F  
0 1 0 rg  
100 100 70 70 re  
F
```

# Content Stream



- Inline Image Extraction
  - Duplications of inline image among content streams
- Content stream operators re-ordering

```
1 0 0 rg
100 100 50 50 re
F
0 0 1 rg
300 300 70 70 re
F
1 0 0 rg
200 200 50 50 re
F
```



```
1 0 0 rg
100 100 50 50 re
200 200 50 50 re
F
0 0 1 rg
300 300 70 70 re
F
```

- Convert format from raw to lossy-compressed (JPEG-like)
  - Hard to perform, as requires PDF specified colorspace to be taken into account
- Image Quality
  - Increase image compression for formats like JPEG



RAW, 1.6MB



JPEG, 0.8MB



JPEG, 30KB

## ■ Image Resolution

- Algorithms: Downsampling, Subsampling, Bicubic Downsampling, etc.
- Keep into account the final scaling on the PDF page



RAW, 1.6MB  
Downscaled on slide



RAW, 50% scaling, 400KB  
Downscaled on slide



RAW, 20% scaling, 64KB  
Upscaled on slide



# Images: Colors



- Color conversions
  - Note: Also can be used outside images
  - Grayscale
  - ColorSpace: 4 components (CMYK) to 3 components (RGB)
  - Smask images with Matte entry

Generally, a detailed discussion of colorspaces is a separate topic

# Images: OCR



- Replace the image with text

- If the image to OCR contains some graphics, then the large image can be replaced with the small ones and recognized text
- Content stream line arts can be OCR'd too



- Keep in mind that some PDFs may contain text written via graphics for security reasons, e.g. to prevent text extraction

## ■ Subsetting

- Cleaning Font Program from unused glyphs
- Not safe for fonts used in forms with user input text



# Fonts: Merging



- Identical font files used in different PDF fonts
  - Font descriptors with some differences, for example different encodings
- Identical PDF fonts with different encodings
  - Replace 2<sup>nd</sup> font usages with the 1<sup>st</sup> font usage by encoding mapping
- Subsets of the same font
  - Quite hard to determine whether the two subsets belongs to the same font and the same **version** of the font

# Fonts: Replacing



- Encoding
  - Replace 2 byte encoding with 1 byte encoding
- Visually similar fonts
  - Replace several visually similar fonts with the minimal single one (either already present in the PDF or a new one)
  - Comparison via the full font or only used glyphs in PDF
  - PANOSE classification
  - Visual appearance may change
- Remove embedded fonts files for standard PDF fonts
  - Losing PDF/A compatibility

# Ordering



1. Flattening
2. Content Streams cleanup
3. Images optimizations
4. Fonts optimization
  1. Replacing
  2. Merging
  3. Subsetting
5. Redundant PDF structures
6. Compressions

- Most of the significant size optimizations usually come from images and fonts resources optimizations
- It is worth checking the size change for any step
  - Some steps may produce heavier file than before
    - Example: images after compression might take more space than the original
  - Some steps may produce little profit with a significant decrease in appearance quality

# Thank you!



Optimization demo/sandbox:

[\*https://itextpdf.com/demos/pdf-optimizer\*](https://itextpdf.com/demos/pdf-optimizer)

- Any questions?