working together!

PDF Days Europe 2022 | Berlin

Strategies for Testing PDF Files

Into the Lion’s Den

Me!

- Research Manager at iText
- Father of 2
- Bass player

- @MyMilkedEek
- @iText

Introducing iText Software

working together!

SDK

iText 7 Suite

Best documented and most feature-rich PDF library

Open-source library for PDF generation and management,

- PDF/A, PDF/UA,
- Digital signing,
- Security,
- HTML conversion,
- Redaction, OCR, etc.,

and closed source add-ons for rendering capabilities, advanced data extraction, advanced forms processing, PDF optimization, advanced language script capabilities, Office to PDF conversion and much more.

High convenience tools

iText DITO

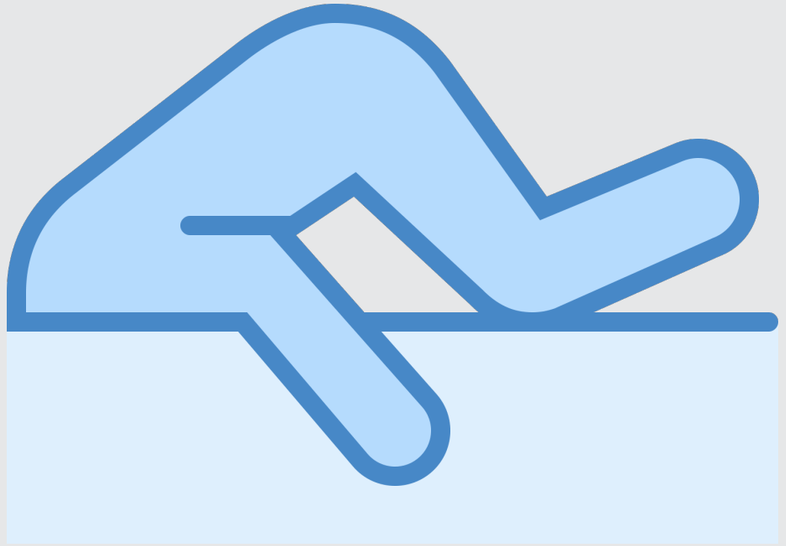
Fast template-based document generation

High convenience technology block – save time
Based on iText technology

- **WYSIWYG Template designer**
- **Conditional logic for content visibility**
- **Data binding**
- **Advanced data visualizations (barcodes/QR codes, charts…)**
- **PDF/UA Compliance Assistance**
- **Template Manager with version, dependency and permission control**
- **Containerized RESTful PDF-producer API (also available as native Java SDK).**

Why test documents?

- Frequently public facing
  - Customers (e.g. Invoices)
  - Citizens <> Governments



Why test documents?

working together!

- Security (exploits)
  - E.g. ZIP bombs
- Compatibility with multiple vendors
- To avoid regression

What needs to be tested?

- Functional requirements
- Visual requirements
- Processing
- Rendering
- Leniency (e.g. HTML)

5 Different Strategies

1. Manual

working together!

Pros

- Before any automation
- Immediate testing
- Large sense of control

Cons

- Slow (corresponds to the project size)
- It looks ok so it probably is > not very precise
- Largely dependent on whoever is testing….

1. Manual

working together!

Pros

- Before any automation
- Immediate testing
- Large sense of control

Cons

- Slow (corresponds to the project size)
- It looks ok so it probably is > not very precise
- Largely dependent on whoever is testing….

1. Manual

Pros

- Before any automation
- Immediate testing
- Large sense of control

Cons

- Slow (corresponds to the project size)
- It looks ok so it probably is > not very precise
- Largely dependent on whoever is testing….

Good for single release, small projects

Strategies <> Metrics

working together!

Strategy	Investment	Ease of Use	Maintainability	Coverage/Test	Focus	Performance
Manual	Low	Low	Low	Med	True	5

2. Auto Visual

Compare appearance to a reference file

Pros

- Very broad visual coverage
- Quick

Cons

- Not testing structure
- No way of knowing if a test fails, exactly why it failed
- If something general changes in the layout, all reference files need to be altered

2. Auto Visual

Compare appearance to a reference file

Pros

- Very broad visual coverage
- Quick

Cons

- Not testing structure
- No way of knowing the exact reason when a test fails
- If something general changes in the layout, all reference files need to be altered

Good when you want your tests to cover a lot visually (in a short time)

Strategies <> Metrics

working together!

Strategy	Investment	Ease of Use	Maintainability	Coverage/Test	Focus	Performance
Manual	Low	Low	Low	Med	True	5
Auto Visual	Med	High	Med	High	False	4

3. Auto Structural

Compare structure to reference file

Pros

- Very precise
- Quick
- Allows differences in order structure wise

Cons

- No way of knowing if a test fails, exactly why it failed
- Breaks easily

3. Auto Structural

Compare structure to reference file

Pros

- Very precise
- Quick
- Allows differences in order structure wise

Cons

- No way of knowing if a test fails, exactly why it failed
- Breaks easily

Good when you want your tests to cover a lot structurally (in a short time)

Strategies <> Metrics

working together!

Strategy	Investment	Ease of Use	Maintainability	Coverage/Test	Focus	Performance
Manual	Low	Low	Low	Med	True	5
Auto Visual	Med	High	Med	High	False	4
Auto Structural	Med	High	Low	High	False	3

4. Byte Comparison

Compare exactly matches reference file

Pros

- Very precise
- Quick

Cons

- Allows no changes in order structure wise
- No way of knowing if a test fails, exactly why it failed
- Extremely fragile

4. Byte Comparison

Compare exactly matches reference file

Pros

- Very precise
- Quick & easy

Cons

- Allows no changes in order structure wise
- No way of knowing if a test fails, exactly why it failed
- Extremely fragile

Good when you want your tests to cover a lot and maintenance is not an issue

Strategies <> Metrics

working together!

Strategy	Investment	Ease of Use	Maintainability	Coverage/Test	Focus	Performance
Manual	Low	Low	Low	Med	True	5
Auto Visual	Med	High	Med	High	False	4
Auto Structural	Med	High	Low	High	False	3
Byte Comparison	Med	High	Low	High	False	1

5. Aspect Based

Check certain aspects with or without a reference file

Pros

- Very focused
- Quick
- Can handle changes in layout, structure

Cons

- Takes a lot of time to create

5. Aspect Based

Check certain aspects with or without a reference file

Pros

- Very focused
- Quick
- Can handle changes in layout, structure

Cons

- Takes a lot of time to create

Good when you have time and your requirements are well defined

Strategies <> Metrics

working together!

Strategy	Investment	Ease of Use	Maintainability	Coverage/Test	Focus	Performance
Manual	Low	Low	Low	Med	True	5
Auto Visual	Med	High	Med	High	False	4
Auto Structural	Med	High	Low	High	False	3
Byte Comparison	Med	High	Low	High	False	1
Aspect Based	High	Med	High	Low	True	2

More . . . Unconventional Methods

Introducing pdfScannerizer

working together!

- Converts an HTML file into a *scanned* PDF
  - Even includes the “Dirty Scanner Hair”

Introducing pdfScannerizer

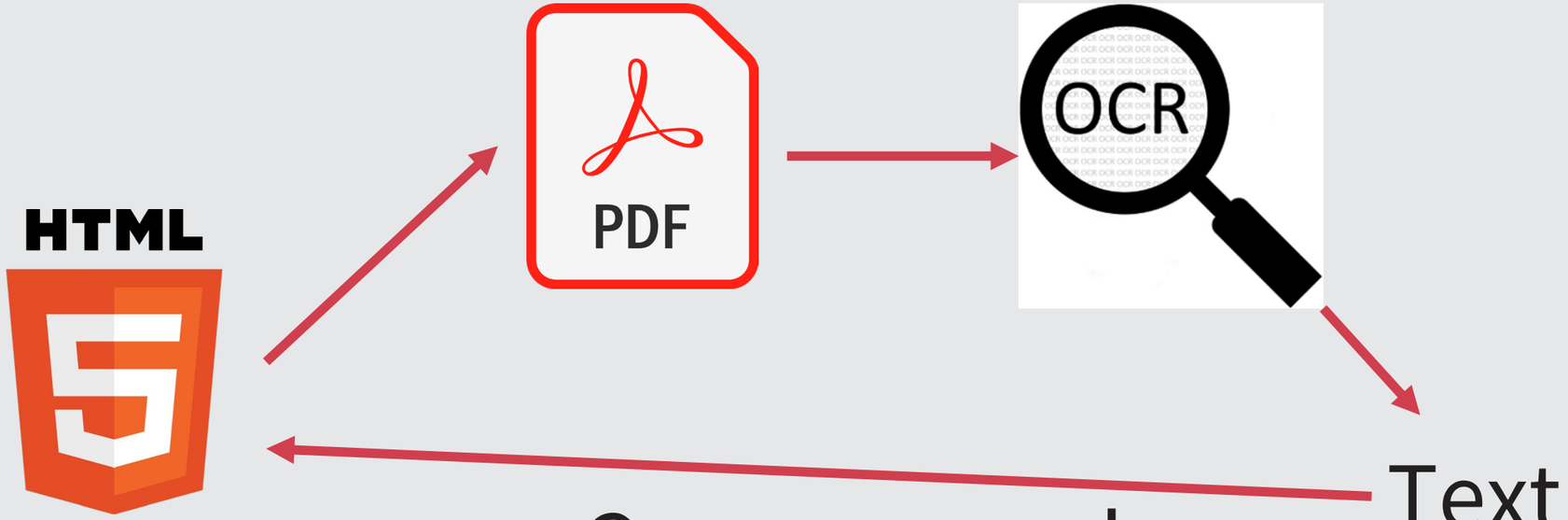
working together!

- Converts an HTML file into a *scanned* PDF
  - Even includes the “Dirty Scanner Hair”

- To benchmark OCR engines

pdfScannerizer

working together!



Arlington Model

- PDF Association resource
  - Check it out on Github
- Set of machine-readable rules
  - What is allowed in dictionaries, which values, which keys are required, etc
- Prototype that converts Arlington Objects into Java objects
  - Allows for **validation of dictionaries**
  - Low cost of maintenance in the testing library
    - Managed through .tsv files

Arlington Model #2

- Arlington could be used as a testing resource
  - Automation of testing your parser using simpler means
  - No need to create whole PDF files by hand
    - A description in Arlington objects should be more than enough

Content Stream "*compilation*"

- "pdfCop"
  - Grammar file based on the spec
    - Not our implementation
  - "Compiles" content stream syntax into an **Abstract Syntax Tree**
    - Validates the syntax and objects
    - Allows for easy inspection of the tree

Conclusions

Conclusions

working together!

- There isn't a "best" strategy
- Possibly combine them
- Choose your strategy based upon
  - Size of the Project
  - Number of releases
  - Metrics
  - (Document type)

Strategy	Investment	Ease of Use	Maintainability	Coverage/Test	Focus	Performance
Manual	Low	Low	Low	Med	True	5
Auto Visual	Med	High	Med	High	False	4
Auto Structural	Med	High	Low	High	False	3
Byte Comparison	Med	High	Low	High	False	1
Aspect Based	High	Med	High	Low	True	2

e.g.

- One release, small project > Manual
- Multiple releases, big project > Aspect based

- Consider GDPR, security and different tools & platforms