

Automated PDF Testing

Carsten Sientop, PDFUnit.com



In the next 45 minutes

- Why tests?
- What to test?
- How to test – PDFUnit's Syntax
- Testing Visible Contents
- Testing Invisible Contents
- PDFUnit–Monitor
- PDFUnit in .NET, Perl, XML



In the next 45 minutes

- Why tests?
- What to test?
- How to test – PDFUnit's Syntax
- Testing Visible Contents
- Testing Invisible Contents
- PDFUnit–Monitor
- PDFUnit in .NET, Perl, XML

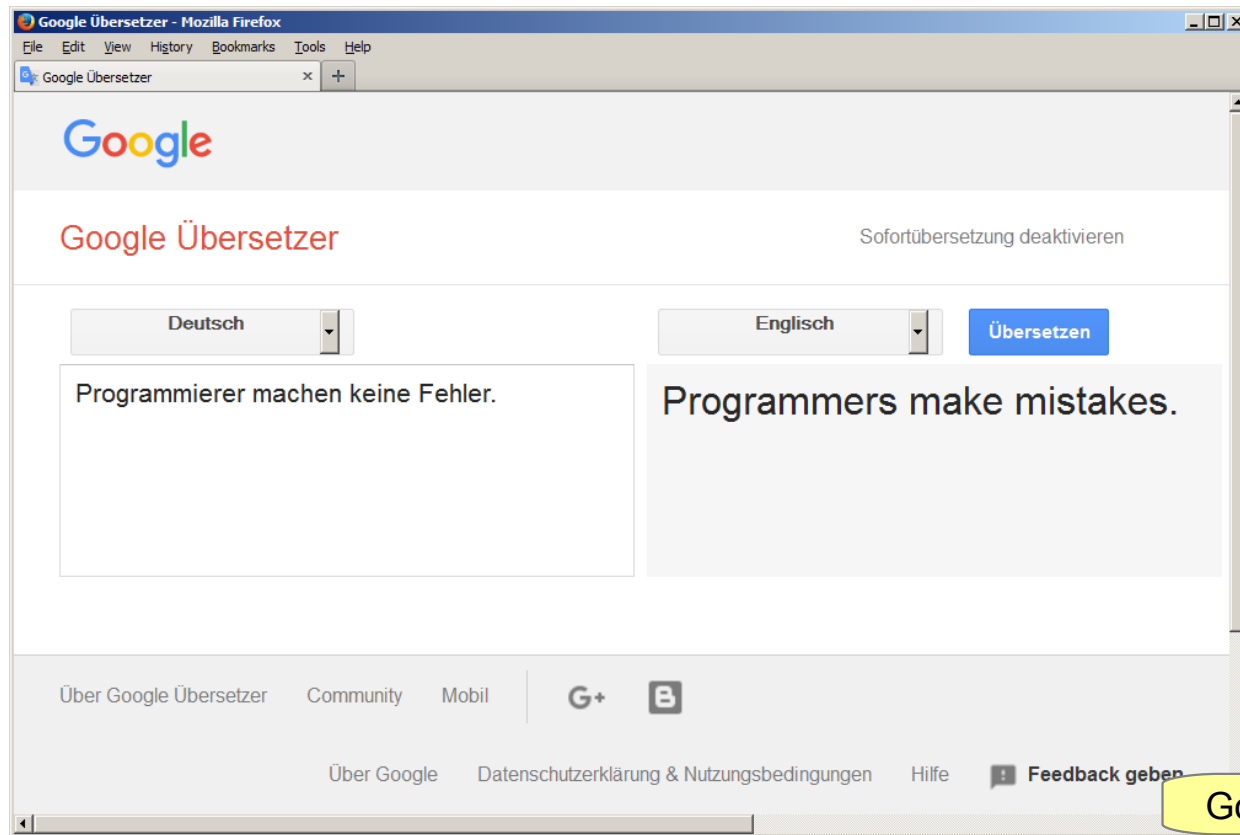


Why tests?

- A PDF is the result of many business processes
- Faulty PDF documents are bad for business
- A PDF generating software contains errors



Why tests? Programmers make mistakes!



Why tests?

- A PDF is the result of many business processes
- Faulty PDF documents are bad for business
- A PDF generating software contains errors
- Programmers make mistakes
- Programs will change
- Computing environment will change



Why tests?

- A PDF is the result of many business processes
- Faulty PDF documents are bad for business
- A PDF generating software contains errors

Testing is expensive

- Programmers make mistakes
- Programs will change
- Computing environment will change



Why tests?

- A PDF is the result of many business processes
- Faulty PDF documents are bad for business
- A PDF generating software contains errors

Testing is expensive

- Programmers make mistakes
- Programs will change
- Computing environment will change

Not testing is more expensive



Why automated tests? (1/2)

- Disadvantages of manual testing:
 - limited human resources
 - results not always reproducible
 - expensive
- Disadvantages of automated testing:
 - It takes time to know the tools



Why automated tests? (2 / 2)

- Advantages of automated testing:
 - no human resources needed
 - directly runnable
 - runnable at any time
 - available for many years
 - inexpensive



What to test?



What to test? Some examples:



XING^x Events

Order information: Order number: 0000000000, Order date: 08 Jun 2016, Ticket number: 13 F 1

Contact for further enquiries: XING Events GmbH, Hans-Bredow-Platz 1, 10245 Berlin, Germany

Event organizer: Association for Digital Elements e.V., Neuenhofer Weg 1, 14057 Berlin, Germany, UST-ID: DE201188888

Your ticket for PDF Days Europe 2016 **1**

Dear Eventbillerbeholder,

Please find below your ticket for 'PDF Days Europe 2016' from 13 June 2016 until 15 June 2016, which you ordered on 8 June 2016. And print out your ticket and take it with you to the event. Please do not hesitate to contact XING Events support if supporting-events.com in case of any problems with your order.

Best regards,
Your XING Events Team

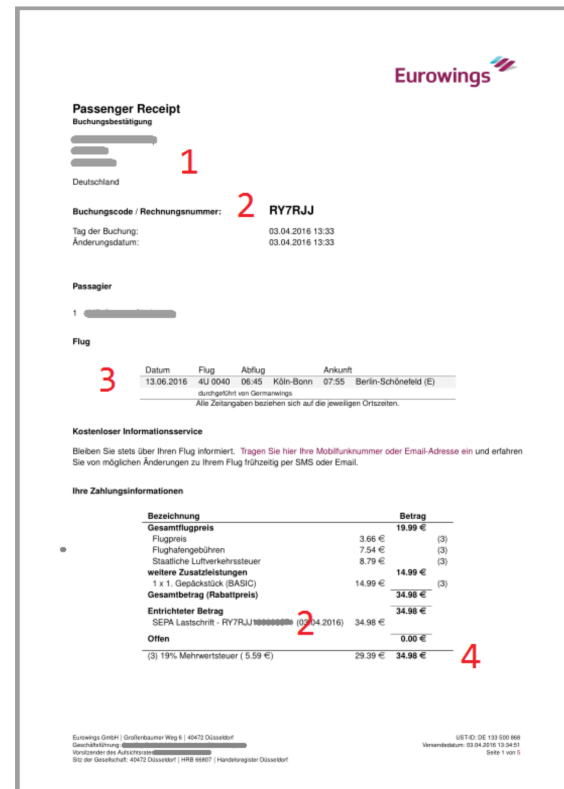
Visit the number no. 1 marketplace for business events. **XING^x Events**

TICKET issued by XING^x Events **2**

PDF Days Europe 2016 **1**

13 June 2016 09:00 until 15 June 2016 18:00
01,5 Campus Berlin
Kastanienallee 82, 10455 Berlin, Germany **3**

PDF Days Europe 2016 - Vendor Ticket
*****VENDOR*****



Eurowings

Passenger Receipt
Buchungsbestätigung **1**

Deutschland

Buchungscode / Rechnungsnummer: **2** RY7RJJ

Tag der Buchung: 03.04.2016 13:33
Änderungsdatum: 03.04.2016 13:33

Passagier **1**

Flug **3**

Datum	Flug	Abflug	Ankunft
13.06.2016	4U 0040	06:45 Köln-Bonn	07:55 Berlin-Schönefeld (E)




durchgeführt von Germanwings
Alle Zeitangaben beziehen sich auf die jeweiligen Ortszeiten.

Kostenloser Informationsservice

Bleiben Sie stets über Ihren Flug informiert. Tragen Sie hier Ihre Mobilfunknummer oder Email-Adresse ein und erfahren Sie von möglichen Änderungen zu Ihrem Flug frühzeitig per SMS oder Email.

Ihre Zahlungsinformationen

Bezeichnung	Betrag
Gesamtflygpreis	19,99 €
Flugpreis	3,66 € (3)
Flughafengebühren	7,54 € (3)
Staatliche Luftverkehrssteuer	8,79 € (3)
weitere Zusatzleistungen	14,99 € (3)
1 x 1. Gepäckstück (BASIC)	14,99 €
Gesamtbetrag (Rabattpreis)	34,98 €
Entrichteter Betrag	34,98 €
SEPA Lastschrift - RY7RJJ0000000000 (03.04.2016)	34,98 €
Offen	-0,00 €
(3) 19% Mehrwertsteuer (5,59 €)	29,39 €
	34,98 € 4

Eurowings GmbH | Großbohrner Weg 1 | 40472 Düsseldorf
Geschäftsführung: 
Vorstandler des Aufsichtsrats: 
Stellv. der Geschäftsführung:  | Handelsregister: Düsseldorf

UST-ID: DE 133 000 888
Versanddatum: 03.04.2016 13:33:41
Seite 1 von 5

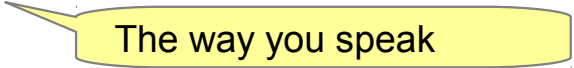


How to test? PDFUnit's Syntax



PDFUnit's Syntax – Single Document

- Dear IT–Jim,
please assert that the document has the new logo.
Best regards, Business–Bill

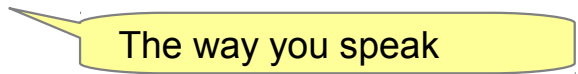


The way you speak



PDFUnit's Syntax – Single Document

- Dear IT–Jim,
please **assert that the document has the new logo.**
Best regards, Business–Bill

The way you speak

PDFUnit's Syntax – Single Document

- Dear IT–Jim,

assert that document has new logo.

Best regards, Business–Bill

The way you speak



PDFUnit's Syntax – Single Document

- Dear IT–Jim,

assert that document has new logo.
Best regards, Business–Bill

The way you speak

The way PUnit speaks

- `AssertThat.document("document.pdf")`
 `.hasImage()`
 `.matching("newLogo.png")`



PDFUnit's Syntax – Multiple Documents

- Dear IT–Jim,
please assert that each document in the folder has
the new logo.
Best regards, Business–Bill

More than one document



PDFUnit's Syntax – Multiple Documents

- Dear IT–Jim,
please **assert that each document in the folder has the new logo.**
Best regards, Business–Bill

More than one document



PDFUnit's Syntax – Multiple Documents

- Dear IT–Jim,
 assert that each document in folder has
 new logo.
Best regards, Business–Bill



PDFUnit's Syntax – Multiple Documents

- Dear IT–Jim,

 assert that each document in folder has
 new logo.
Best regards, Business–Bill
- AssertThat.eachDocument()
 .inFolder(..)
 .hasImage()
 .matching("newLogo.png")

Example 101



PDFUnit's Syntax – Comparing Two Documents

- Dear IT–Jim,

please assert that the PDF under test and the reference PDF have the same number of pages.
Best regards, Business–Bill



PDFUnit's Syntax – Comparing Two Documents

- Dear IT–Jim,

please **assert that** the PDF under test **and** the reference PDF **have the same number of pages**.
Best regards, Business–Bill



PDFUnit's Syntax – Comparing Two Documents

- Dear IT–Jim,

please **assert that the PDF under test and the reference PDF have the same number of pages.**

Best regards, Business–Bill



PDFUnit's Syntax – Comparing Two Documents

- Dear IT–Jim,

assert that PDF under test and
reference PDF have same number of pages.

Best regards, Business–Bill



PDFUnit's Syntax – Comparing Two Documents

- Dear IT–Jim,

assert that PDF under test and
reference PDF have same number of pages.

Best regards, Business–Bill

- `AssertThat.document(pdfUnderTest)`
 `.and(referencePDF)`
 `.haveSameNumberOfPages()`

Example 102



Page Selection – Using Constants

- AssertThat.document(..)
 - .restrictedTo(FIRST_PAGE)
 - .hasText()
- Predefined Constants:
 - FIRST_PAGE
 - LAST_PAGE
 - EVERY_PAGE
 - ANY_PAGE
 - EVEN_PAGES
 - ODD_PAGES



Page Selection – Individual Pages and Page Ranges

- ... `.restrictedTo(page3)`
... `.restrictedTo(pages123)`
... `.restrictedTo(pages1To3)`
- `page3` = `PagesToUse.getPage(3);`
`pages123` = `PagesToUse.getPages(1, 2, 3);`
`pages1To3` = `PagesToUse.spanningFrom(1).to(3);`

All requirements are covered



Page Regions

- ... `.restrictedTo(bodyRegion)`
... `.restrictedTo(addressRegion)`
- `int leftX = 18;`
`int upperY = 30;`
`int w = 182;`
`int h = 238;`
`bodyRegion = new PageRegion(leftX, upperY, w, h);`
- `addressRegion = new PageRegion(..);`

x=0, y=0 is the upper left corner



The First Example

```
public void hasTextSpanningOver2Pages() {
    String filename = "document-under-test.pdf";
    String textOnPage1 = "Text starts on page 1 and ";
    String textOnPage2 = "continues on page 2";
    String expectedText = textOnPage1 + textOnPage2;
    PagesToUse pages1to2 = PagesToUse.spanningFrom(1).to(2);

    int leftX = 18;
    int upperY = 30;
    int width = 182;
    int height = 238;
    PageRegion bodyRegion = new PageRegion(leftX, upperY, width, height);

    AssertThat.document(filename)
        .restrictedTo(pages1to2)
        .restrictedTo(bodyRegion)
        .hasText()
        .containing(expectedText)
    ;
}
```

Example 103



Testing Visible Contents



Testing Visible Contents

- Text in pages and page regions
- Ordered text
- Rendered pages, rendered regions
- Images
- Form fields
- Bookmarks
- Bar code, QR code
- Text in images (OCR)
- Right-to-left text

Example 201

Example 202

Example 203

Example 204

Example 205

Example 206

Example 207

Example 208

Example 209

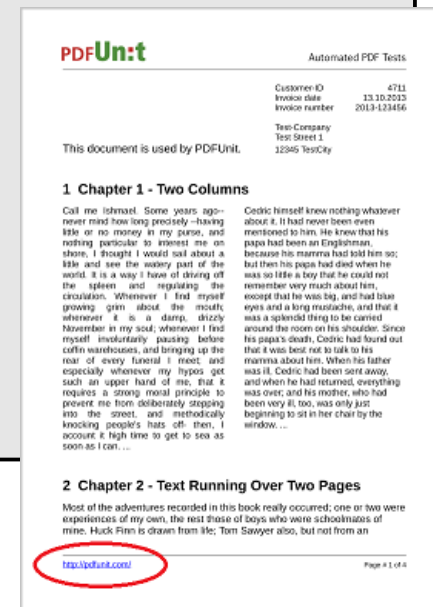
Skip examples



Example – Text in Page Regions

```
@Test
public void hasLinkInFooterAfterPage1 () {
    String pdfUnderTest = "document-under-test.pdf";
    String linkToHomepage = "http://pdfunit.com/";
    PagesToUse pagesAfter1 = ON_EVERY_PAGE.after(1);
    PageRegion footerRegion = createFooterRegion();

    AssertThat.document(pdfUnderTest)
        .restrictedTo(pagesAfter1)
        .restrictedTo(footerRegion)
        .hasText()
        .containing(linkToHomepage)
        ;
}
```



↑ overview

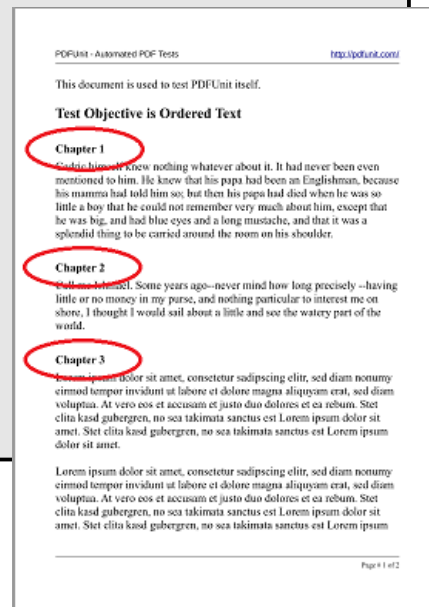
Example 201



Example – Ordered Text

```
@Test
public void hasTextInOrder() {
    String pdfUnderTest = "document-under-test.pdf";
    String titleChapter1 = "Chapter 1";
    String titleChapter2 = "Chapter 2";
    String titleChapter3 = "Chapter 3";

    AssertThat.document(pdfUnderTest)
        .hasText()
        .first(titleChapter1)
        .then(titleChapter2)
        .then(titleChapter3)
        ;
}
```



↑ overview

Example 202



Example – Compare Rendered Pages

```
@Test
public void haveSameAppearanceInRegion() {
    String pdfUnderTest = "document-under-test.pdf";
    String pdfReference = "reference.pdf";

    int leftX = 0; // millimeters
    int upperY = 0;
    int width = 210;
    int height = 50;
    PageRegion pageRegion = new PageRegion(leftX, upperY, width, height);

    AssertThat.document(pdfUnderTest)
        .and(pdfReference)
        .restrictedTo(EVERY_PAGE)
        .restrictedTo(pageRegion)
        .haveSameAppearance();
}
```

[↑ overview](#)[Example 203](#)

Example – Check Images

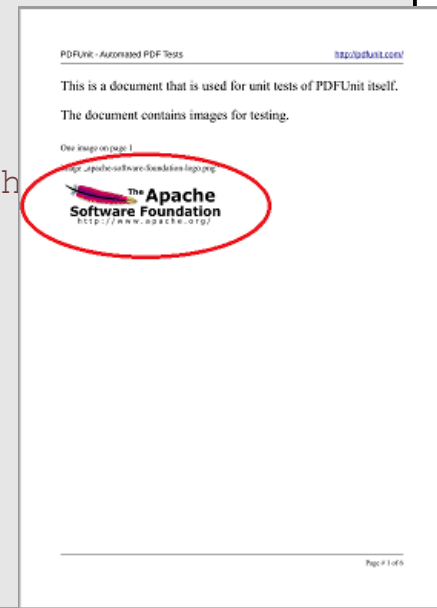
```

@Test
public void hasImageInRegion() {
    String pdfUnderTest = "document-under-test.pdf";
    String logoASF = "images/apache-software-foundation-logo.png";

    int leftX = 15; // millimeters
    int upperY = 80;
    int w = 150;
    int h = 45;
    PageRegion logoRegion = new PageRegion(leftX, upperY, w, h);
    PagesToUse page3 = PagesToUse.getPages(3);

    AssertThat.document(pdfUnderTest)
        .restrictedTo(page3)
        .restrictedTo(logoRegion)
        .hasImage()
        .matching(logoASF)
    ;
}

```



[↑ overview](#)

[Example 204](#)



Example – Form Fields

```
@Test
public void allFieldsWithoutTextOverflow() {
    String pdfUnderTest = "document-under-test.pdf";

    AssertThat.document(pdfUnderTest)
        .hasFields()
        .allWithoutTextOverflow();
}
```

This is a document used for unit tests of PDFUnit itself.
It is generated with iText by 'CreatePDF_FieldSizeAndText.java'.

The fields in this document are used to test whether the content fits inside or not.

Textfield, text inside, align left:	
Textfield, text inside, align right:	
Textfield, too much text, align left:	
Textfield, too much text, align right:	
Textfield, too much text, multiline:	

[↑ overview](#)

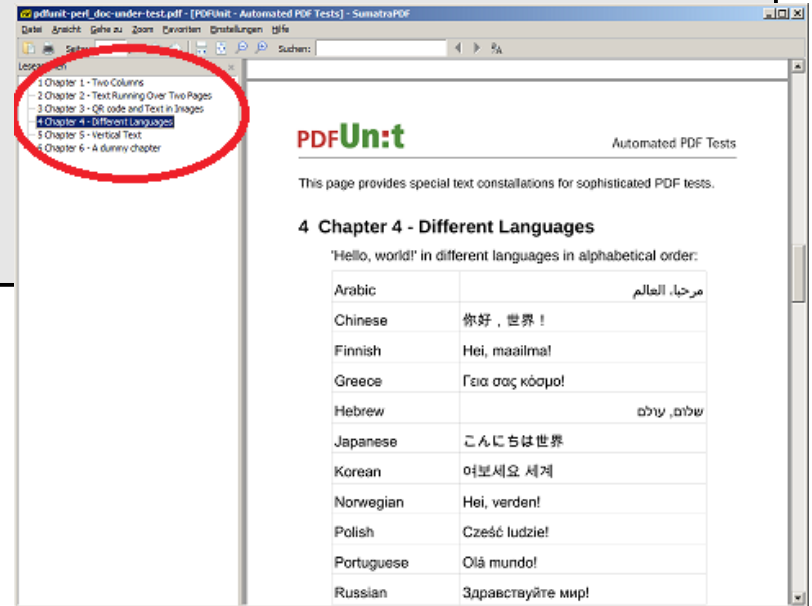
[Example 205](#)



Example – Bookmarks

```
@Test
public void hasBookmarkWithLabel() {
    String pdfUnderTest = "document-under-test.pdf";
    String bookmarkChapter4 = "4 Chapter 4 - Different Languages";

    AssertThat.document(pdfUnderTest)
        .hasBookmark()
        .withLabel(bookmarkChapter4)
    ;
}
```



↑ overview

Example 206



Example – Bar Code, QR Code

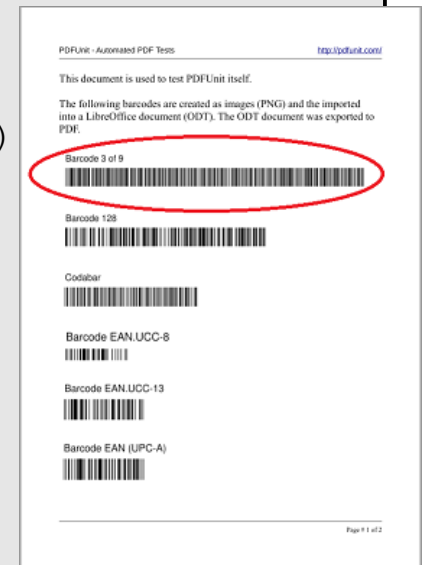
```
@Test
public void hasBarCode_3of9() {
    String pdfUnderTest = "document-under-test.pdf";

    int leftX    = 0; // millimeters
    int upperY   = 70;
    int w        = 210;
    int h        = 30;
    PageRegion pageRegion = new PageRegion(leftX, upperY, w, h)

    AssertThat.document(pdfUnderTest)
        .restrictedTo(FIRST_PAGE)
        .restrictedTo(pageRegion)
        .hasImage()
        .withBarcode()
        .containing("TESTING BARCODES WITH PDFUNIT")
    ;
}
```

↑ overview

Example 207



Example – Text in Images (OCR)

```
@Test
public void validateTextInImages() {
    String pdfUnderTest = "document-under-test.pdf";
    String expectedText = "S4NM3G";
    PagesToUse page2 = PagesToUse.getPage(2);
    PageRegion imageRegion = createOCRImageRegion();

    AssertThat.document(pdfUnderTest)
        .restrictedTo(page2)
        .restrictedTo(imageRegion)
        .hasImage()
        .withText()
        .equalsTo(expectedText)
    ;
}
```

This is a PNG image:



[↑ overview](#)

[Example 208](#)



Example – RTL Text (right-to-left)

```
@Test
public void validateRightToLeftText() {
    String pdfUnderTest = "document-under-test.pdf";
    String hello_ar = "مرحبا، العالم";
    String hello_cn = "你好，世界！";
    String hello_he = "שלום, עולם";
    String hello_ru = "Здравствуйтe мир!";
    PagesToUse page3 = PagesToUse.getPage(3);

    AssertThat.document(pdfUnderTest)
        .restrictedTo(page3)
        .hasText()
        .containing(hello_ar)
        .containing(hello_cn)
        .containing(hello_he)
        .containing(hello_ru)
    ;
}
```

[↑ overview](#)

[Example 209](#)

4 Chapter 4 - Different Languages

'Hello, world!' in different languages in alphabetical order:

Arabic	مرحبا، العالم
Chinese	你好，世界！
Hebrew	שלום, עולם
Russian	Здравствуйтe мир!



Testing Invisible Contents



Testing Invisible Contents

- Hidden text
- Empty regions
- Field properties
- Contents of ZUGFeRD data
- Compliance with PDF/A-1
- Compliance with design rules
- Signature data (signed by, ...)
- Fonts
- JavaScript

Example 301

Example 302

Example 303

Example 304

Example 305

Example 306

Example 307

Example 308

Example 309

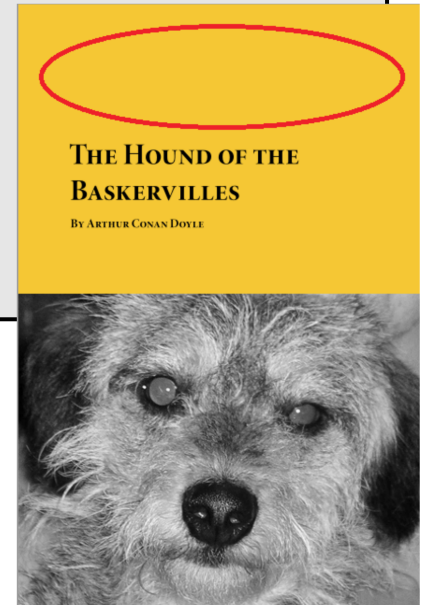
Skip examples



Example – Hidden Text

```
@Test
public void checkInvisibleText() {
    String pdfUnderTest = "the-hound-of-baskervilles.pdf";
    String expectedText = "Download free eBooks of classic";

    AssertThat.document(pdfUnderTest)
        .restrictedTo(FIRST_PAGE)
        .hasText()
        .startingWith(expectedText)
    ;
}
```



[↑ overview](#)

[Example 301](#)



Example – Empty Regions

```
@Test
public void lastPageBodyShouldBeEmpty() {
    String pdfUnderTest = "document-under-test.pdf";
    PageRegion textBody = createBodyRegion();

    AssertThat.document(pdfUnderTest)
        .restrictedTo(LAST_PAGE)
        .restrictedTo(textBody)
        .hasNoImage()
        .hasNoText();
}
```

[↑ overview](#)[Example 302](#)

Example – Form Field Properties

```
@Test
public void hasField_Exportable() throws Exception {
    String pdfUnderTest = "document-under-test.pdf";
    String fieldNameExportableField = "ageField";

    AssertThat.document(filename)
        .hasField(fieldNameExportableField)
        .withProperty()
        .exportable()
        ;
}
```

PDFUnit - Automated PDF Tests

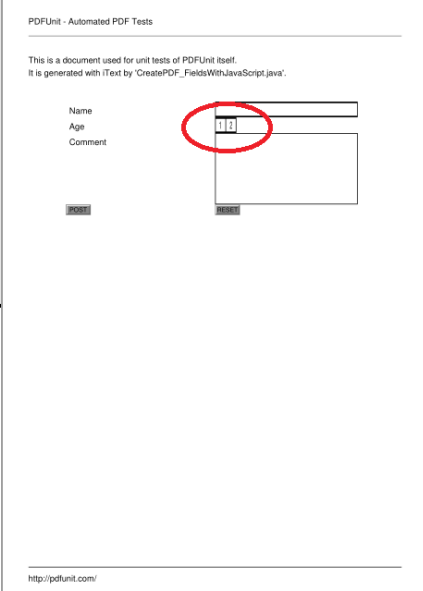
This is a document used for unit tests of PDFUnit itself.
It is generated with iText by 'CreatePDF_FieldsWithJavaScript.java'.

Name

Age

Comment

<http://pdfunit.com/>

A screenshot of a PDF form titled "PDFUnit - Automated PDF Tests". The form contains a header with a title and a subtitle, followed by three text input fields labeled "Name", "Age", and "Comment". Below the input fields are two buttons labeled "POST" and "RESET". A red circle is drawn around the "Age" input field. At the bottom of the page, there is a URL "http://pdfunit.com/".

[↑ overview](#)

[Example 303](#)



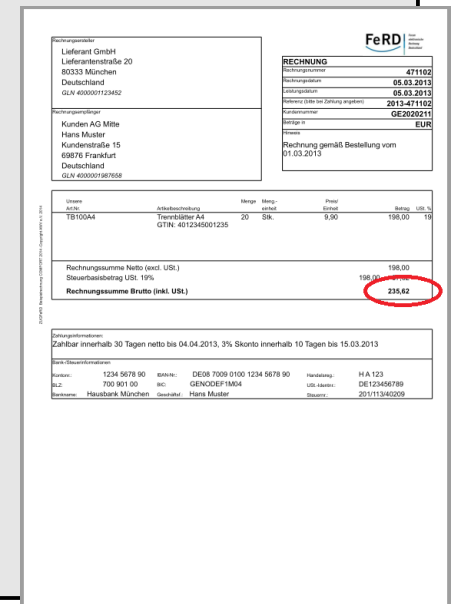
Example – Contents in ZUGFeRD data

```

public void validateTotalSum() {
    String pdfUnderTest = "ZUGFeRD_1p0_BASIC_Einfach.pdf";
    String expectedTotalXML = "235.62";
    String expectedTotalPDF = "235,62";
    XMLNode nodeTotal = new XMLNode("ram:GrandTotalAmount", expectedTotalXML);
    PageRegion regionTotal = createRegionTotal();

    AssertThat.document(pdfUnderTest)
        .hasZugferdData()
        .withNode(nodeTotal)
        ;
    AssertThat.document(pdfUnderTest)
        .restrictedTo(FIRST_PAGE)
        .restrictedTo(regionTotal)
        .hasText()
        .containing(expectedTotalPDF)
        ;
}

```



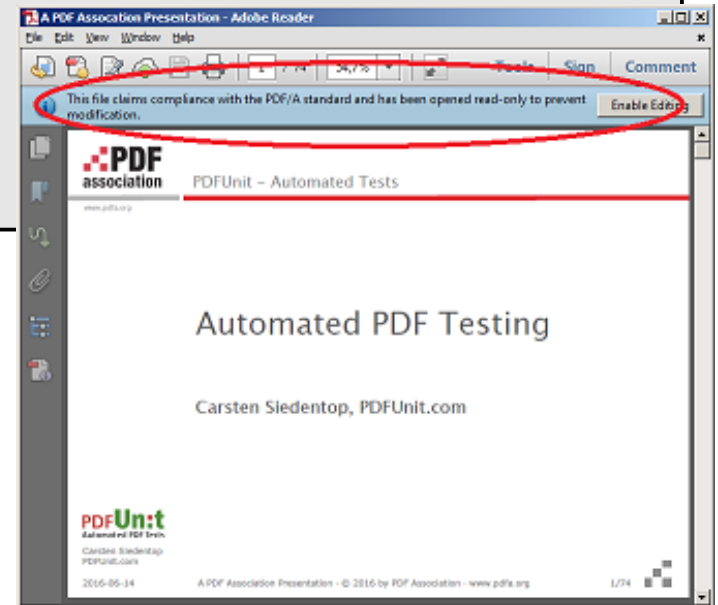
↑ overview

Example 304

Example – Compliance with PDF/A-1a

```
@Test
public void compliesWithPDFA() {
    String pdfUnderTest = "document-under-test.pdf";

    AssertThat.document(pdfUnderTest)
        .compliesWith()
        .pdfStandard(Standard.PDFA_1A)
    ;
}
```



↑ overview

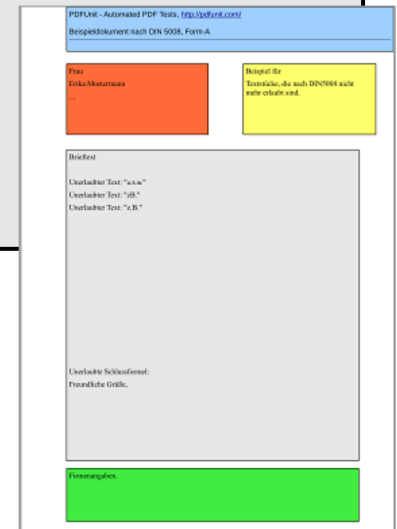
Example 305



Example – Compliance with Company Rules

```
@Test
public void letterCompliesWithDIN5008() {
    String pdfUnderTest = "document-under-test.pdf";
    String rules5008 = "din5008-formA_brief-hochkant.xls";
    PDFValidationConstraints constr = new PDFValidationConstraints(rules5008);

    AssertThat.document(pdfUnderTest)
        .compliesWith()
        .constraints(constr)
        ;
}
```



↑ overview

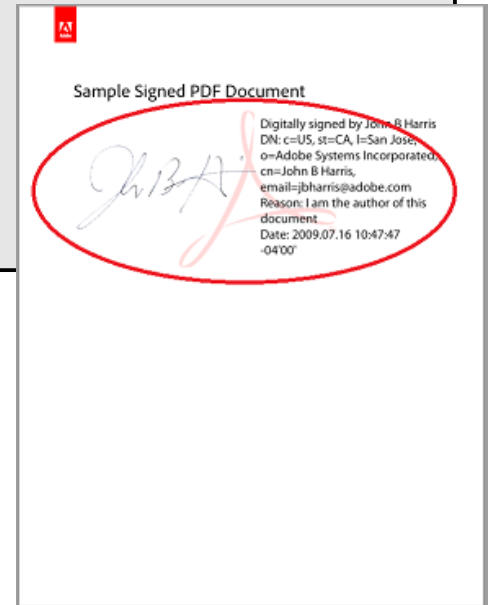
Example 306



Example – Signature Data

```
@Test
public void checkSigningDate() {
    String pdfUnderTest = "document-under-test.pdf";
    Calendar signingDate = DateHelper.getCalendar("2007-10-14", "yyyy-MM-dd");

    AssertThat.document(pdfUnderTest)
        .hasSignatureField("sign_rbl")
        .signedOn(signingDate)
    ;
}
```



[↑ overview](#)

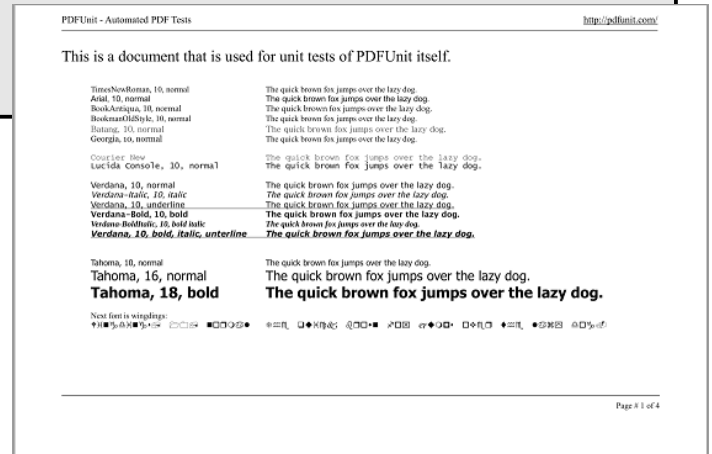
[Example 307](#)



Example – Fonts

```
@Test
public void shouldNotHaveFont_ComicSansMS () {
    String pdfUnderTest = "document-under-test.pdf";
    String fontComic = "ComicSansMS";

    AssertThat.document(pdfUnderTest)
        .hasFont()
        .withNameNotContaining(fontComic)
    ;
}
```



↑ overview

Example 308



Example – JavaScript

```
@Test
public void hasJavaScript_DateFunctions() {
    String pdfUnderTest = "document-under-test.pdf";

    AssertThat.document(pdfUnderTest)
        .hasJavaScript()
        .containing("function DataClassTime")
        .containing("function DataClassTime_Format")
        .containing("function DataClassTime_Scan")
        ;
}
```

Bitte aktivieren Sie JavaScript.

Stadt Köln Die Oberbürgermeisterin Wohnungsgeberbescheinigung nach § 19 des Bundesmietgesetzes

Angaben zur Wohnung
 Herr/n wird ein Einzug in Auszug aus folgender Wohnung bestell/t:
 Straße und Hausnummer (mit Zusatz) Postleitzahl Ort Köln
 Stockwerk, Wohnungsnummer beziehungsweise Lagebeschreibung der Wohnung im Haus
 Datum des Einzuges oder Auszuges

Folgende Personen sind aus- oder eingezogen:
 Familienname Vorname
 Familienname Vorname
 Familienname Vorname
 Familienname Vorname
 Familienname Vorname
 Familienname Vorname

Falls mehr Personen anzugeben sind, bitte Familienname und Vorname hier eintragen
 Familienname Vorname

Name und Anschrift der Wohnungsgeberin oder des Wohnungsgebers:
 Familienname Vorname
 Straße und Hausnummer Postleitzahl Ort
 Gegebenenfalls Name und Vorname der durch den Wohnungsgeber beauftragten Person

Wohnungsgeberin ist gleichzeitig Eigentümerin der Wohnung
 Wohnungsgeberin ist nicht Eigentümerin der Wohnung

Zwischenspeichern Drucken Weiter
 Seite 1 von 2

↑ overview

Example 309

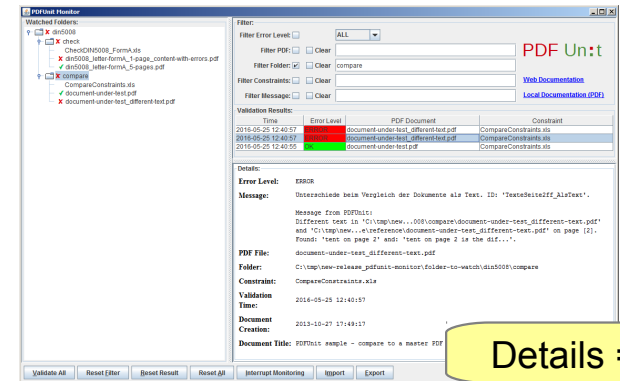


PDFUnit-Monitor



PDFUnit-Monitor

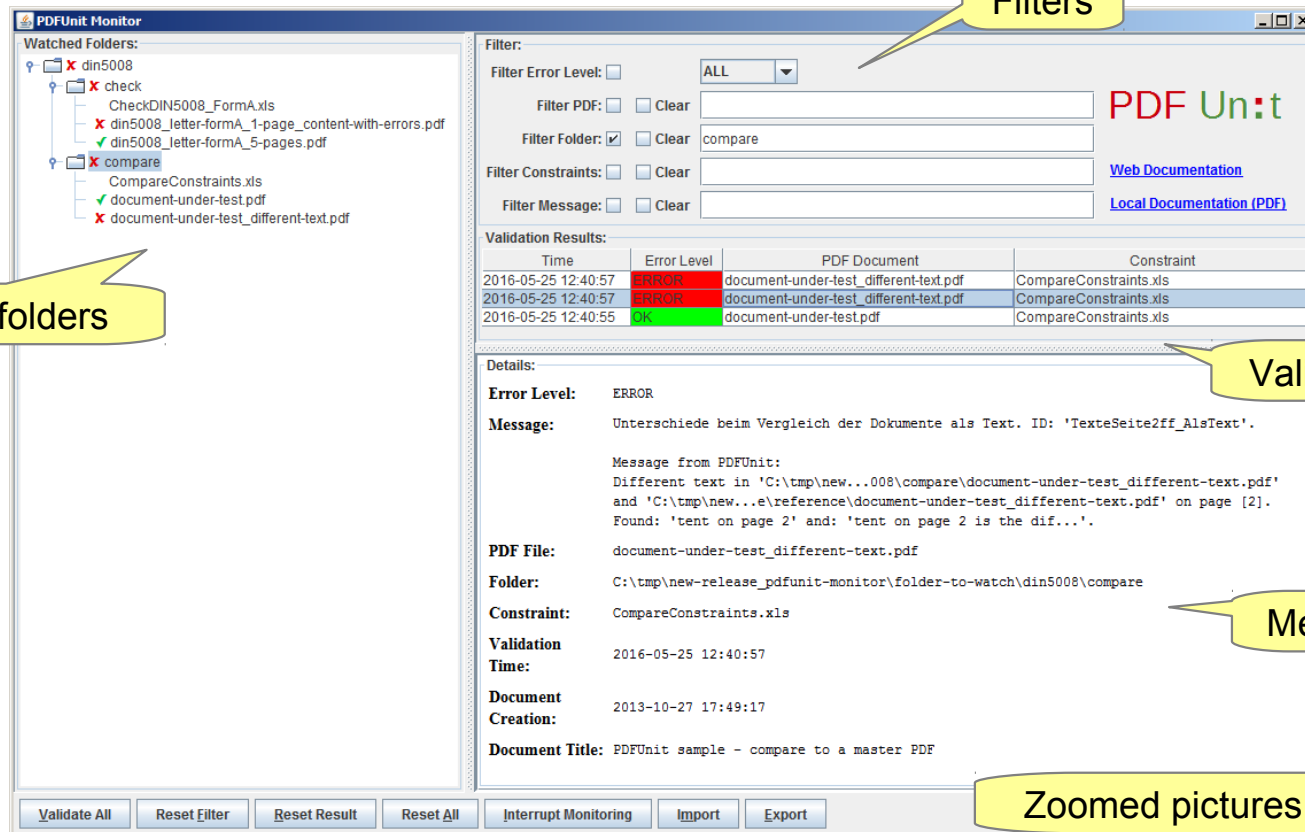
- Continuously watching PDF in folders
- Suitable for non-programmers
- Validation rules are put in Excel files
- Also for interactive usage



Details =>



PDFUnit-Monitor – Continuously Monitoring



The screenshot shows the PDFUnit Monitor application interface. On the left, a tree view shows 'Watched folders' including 'din5008', 'check', and 'compare'. The main area displays 'Filters' (Filter Error Level: ALL, Filter PDF, Filter Folder: compare, Filter Constraints, Filter Message) and 'Validation Results' (a table with columns: Time, Error Level, PDF Document, Constraint). Below the table is a 'Details' section showing 'Error Level: ERROR', 'Message: Unterschiede beim Vergleich der Dokumente als Text. ID: 'TexteSeite2ff_AlsText''.', 'PDF File: document-under-test_different-text.pdf', 'Folder: C:\tmp\new-release_pdfunit-monitor\folder-to-watch\din5008\compare', 'Constraint: CompareConstraints.xls', 'Validation Time: 2016-05-25 12:40:57', 'Document Creation: 2013-10-27 17:49:17', and 'Document Title: PDFUnit sample - compare to a master PDF'. At the bottom, there are buttons for 'Validate All', 'Reset Filter', 'Reset Result', 'Reset All', 'Interrupt Monitoring', 'Import', and 'Export'.

Time	Error Level	PDF Document	Constraint
2016-05-25 12:40:57	ERROR	document-under-test_different-text.pdf	CompareConstraints.xls
2016-05-25 12:40:57	ERROR	document-under-test_different-text.pdf	CompareConstraints.xls
2016-05-25 12:40:55	OK	document-under-test.pdf	CompareConstraints.xls

Filters

Watched folders

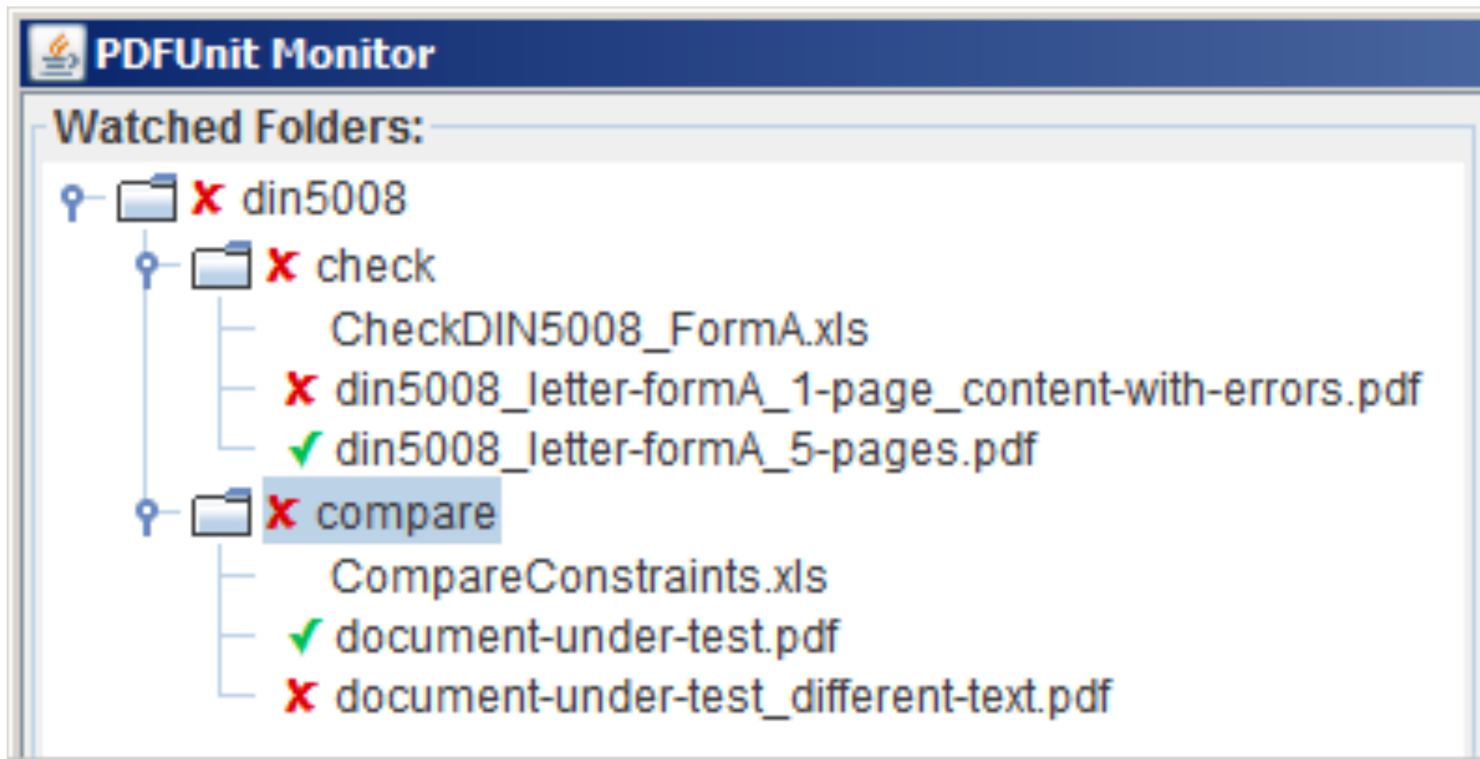
Validation results

Message details

Zoomed pictures are following

Skip them

PDFUnit-Monitor – Watched Folders



PDFUnit-Monitor – Result List and Message Filters

Filter:

Filter Error Level: ALL ▼

Filter PDF:

Filter Folder:

Filter Constraints:

Filter Message:

Validation Results:

Time	Error Level	PDF Document	
2016-05-25 12:40:57	ERROR	document-under-test_different-text.pdf	CompareCon
2016-05-25 12:40:57	ERROR	document-under-test_different-text.pdf	CompareCon
2016-05-25 12:40:55	OK	document-under-test.pdf	CompareCon



PDFUnit-Monitor – Message Details

Details:**Error Level:** ERROR**Message:** Unterschiede beim Vergleich der Dokumente als Text. ID: 'TexteSeite2ff_AlsText'.

Message from PDFUnit:

```
Different text in 'C:\tmp\new...008\compare\document-under-test_different-text.pdf'  
and 'C:\tmp\new...e\reference\document-under-test_different-text.pdf' on page [2].  
Found: 'tent on page 2' and: 'tent on page 2 is the dif...'
```

PDF File: document-under-test_different-text.pdf**Folder:** C:\tmp\new-release_pdfunit-monitor\folder-to-watch\din5008\compare**Constraint:** CompareConstraints.xls**Validation
Time:** 2016-05-25 12:40:57**Document
Creation:** 2013-10-27 17:49:17**Document Title:** PDFUnit sample - compare to a master PDF

Application and messages in English and German



Programming Languages



PDFUnit is available for:

- Java [Example 401](#)
- .NET [Example 402](#)
- Perl [Example 403](#)
- XML [Example 404](#)



Example in Java

```
@Test
public void haveSameAppearanceInRegion() throws Exception {
    String pdfUnderTest = "document-under-test.pdf";
    String pdfReference = "reference.pdf";

    int leftX = 0; // millimeters
    int upperY = 0;
    int width = 210;
    int height = 50;
    PageRegion pageRegion = new PageRegion(leftX, upperY, width, height);

    AssertThat.document(pdfUnderTest)
        .and(pdfReference)
        .restrictedTo(EVERY_PAGE)
        .restrictedTo(pageRegion)
        .haveSameAppearance();
}
```

[↑ overview](#)[Example 401](#)

Example in .NET, C#

```
[TestMethod]
public void haveSameAppearanceInRegion() {
    String pdfUnderTest = "document-under-test.pdf";
    String pdfReference = "reference.pdf";

    int leftX = 0;
    int upperY = 0;
    int width = 210;
    int height = 50;
    PageRegion region = new PageRegion(leftX, upperY, width, height);

    AssertThat.document(pdfUnderTest)
        .and(pdfReference)
        .restrictedTo(Constants.EVERY_PAGE)
        .restrictedTo(region)
        .haveSameAppearance();
}
```

[↑ overview](#)[Example 402](#)

Example in Perl

```
lives_ok {  
    my $pdfUnderTest = "document-under-test.pdf";  
    my $pdfReference = "reference.pdf";  
  
    my $leftX    = 0;  
    my $upperY  = 0;  
    my $width   = 210;  
    my $height  = 50;  
    my $region = PageRegion->new($leftX, $upperY, $width, $height);  
  
    AssertThat->document($pdfUnderTest)  
        ->and($pdfReference)  
        ->restrictedTo(EVERY_PAGE)  
        ->restrictedTo($region)  
        ->haveSameAppearance()  
;  
} "haveSameAppearanceInRegion";
```

[↑ overview](#)[Example 403](#)[Perl-Wrapper by Axel Miesen, http://search.cpan.org/dist/PDF-PDFUnit/](http://search.cpan.org/dist/PDF-PDFUnit/)

Example in XML

```
<testcase name="haveSameAppearanceInRegion">
  <assertThat pdfUnderTest="document-under-test.pdf"
    pdfReference="reference.pdf"
  >
    <haveSameAppearance on="EVERY_PAGE">
      <inRegion upperLeftX="0"
        upperLeftY="0"
        width="210"
        height="50"
      />
    </haveSameAppearance>
  </assertThat>
</testcase>
```

[↑ overview](#)[Example 404](#)

Conclusion



PDFUnit – Conclusion

- Programmers make mistakes (you remember)
- One mistake is not to test



PDFUnit – Conclusion

- Programmers make mistakes (you remember)
- One mistake is not to test
- Another mistake is to accept the situation



PDFUnit – Conclusion

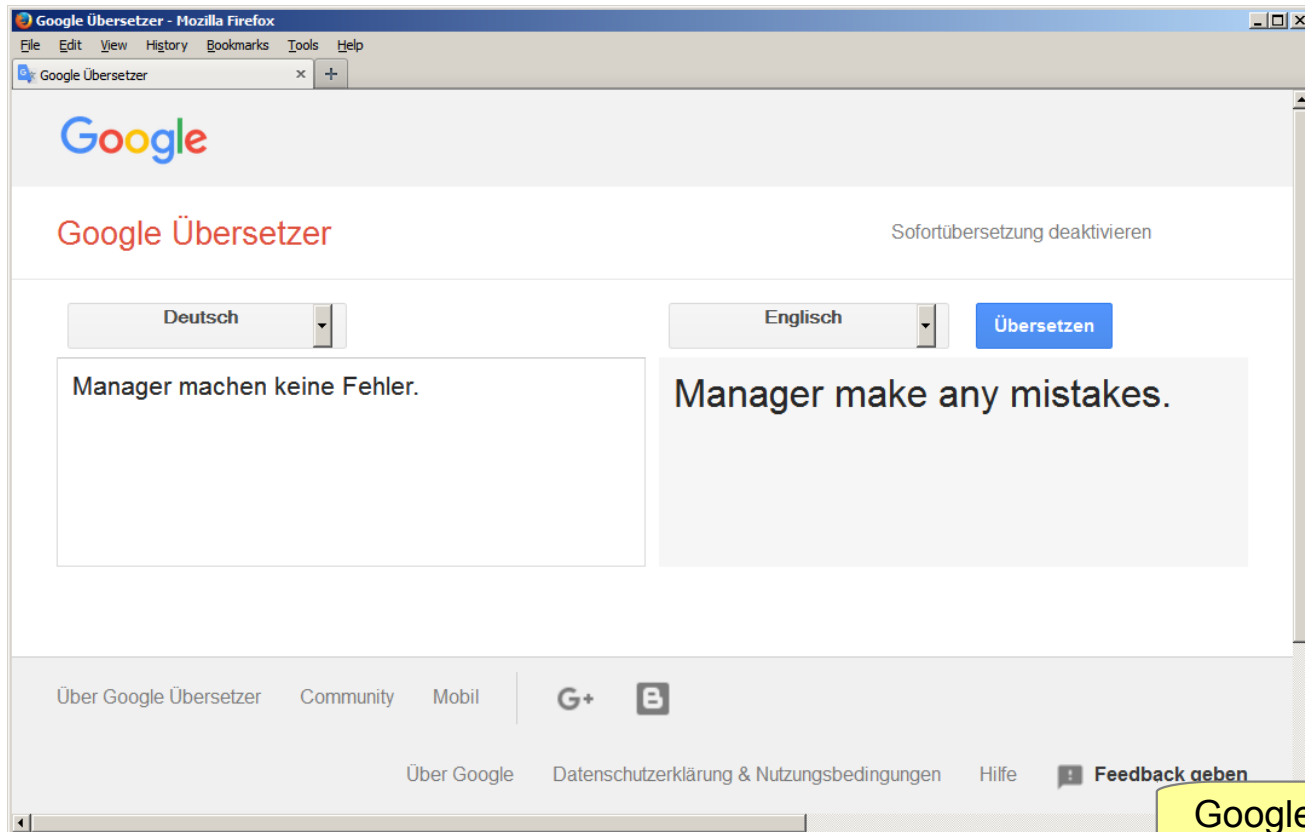
- Programmers make mistakes (you remember)
- One mistake is not to test
- Another mistake is to accept the situation
- The management has to support the decision to invest in automated tests.

Sometimes managers make mistakes (ask Google)

Skip management errors



Managers ...



Testing PDF with PDFUnit is easy

Thank you! Any questions?



Additional Slides



Additional Slides

- Challenging Tests
- PDFUnit is well tested
- PDFUnit is well documented
- PDFUnit at runtime

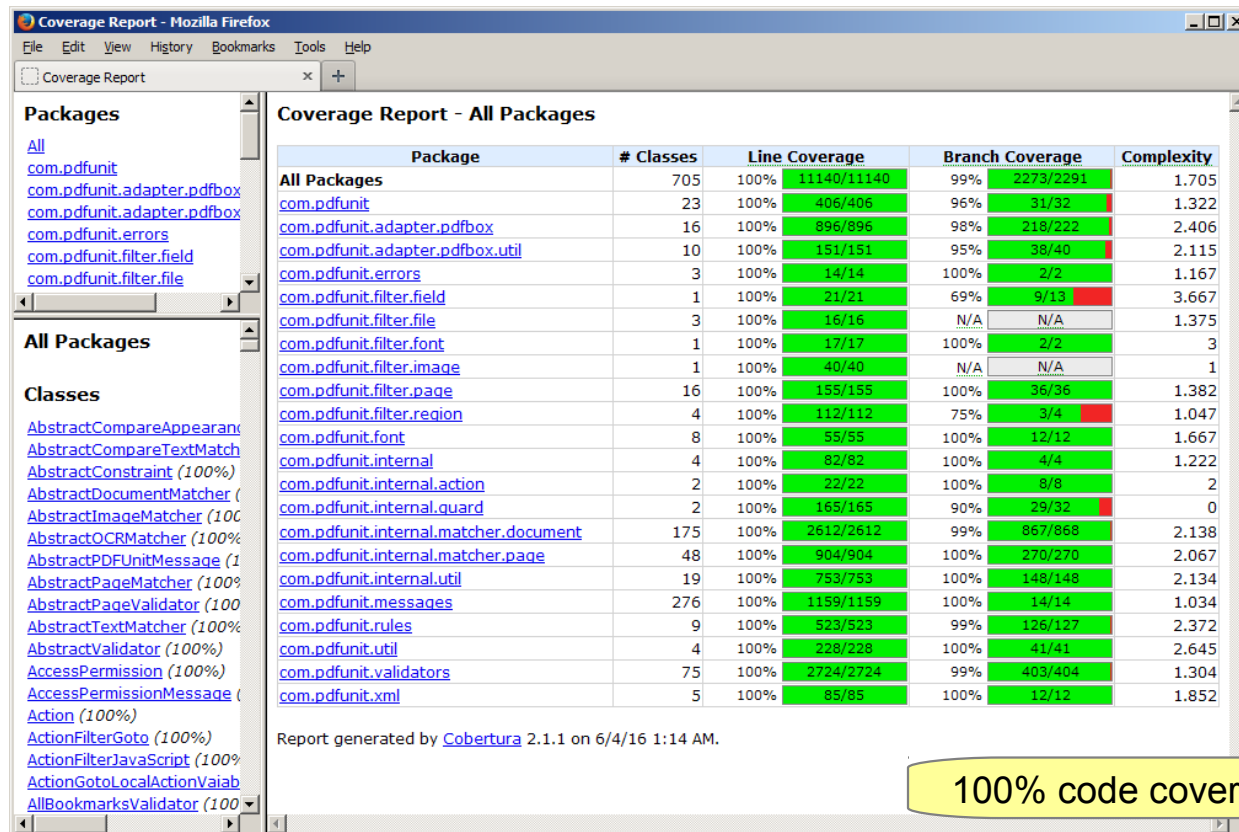


Challenging Test Scenarios

- Create a PDF,
send it by email,
and validate the received attachment Example 501
- Create a PDF by a Web application,
validate the downloaded PDF Example 502
- Compare ZUGFeRD content
with visible PDF data Example 503



PDFUnit is well tested



Coverage Report - All Packages

Package	# Classes	Line Coverage	Branch Coverage	Complexity
All Packages	705	100% 11140/11140	99% 2273/2291	1,705
com.pdfunit	23	100% 406/406	96% 31/32	1,322
com.pdfunit.adapter.pdfbox	16	100% 896/896	98% 218/222	2,406
com.pdfunit.adapter.pdfbox.util	10	100% 151/151	95% 38/40	2,115
com.pdfunit.errors	3	100% 14/14	100% 2/2	1,167
com.pdfunit.filter.field	1	100% 21/21	69% 9/13	3,667
com.pdfunit.filter.file	3	100% 16/16	N/A N/A	1,375
com.pdfunit.filter.font	1	100% 17/17	100% 2/2	3
com.pdfunit.filter.image	1	100% 40/40	N/A N/A	1
com.pdfunit.filter.page	16	100% 155/155	100% 36/36	1,382
com.pdfunit.filter.region	4	100% 112/112	75% 3/4	1,047
com.pdfunit.font	8	100% 55/55	100% 12/12	1,667
com.pdfunit.internal	4	100% 82/82	100% 4/4	1,222
com.pdfunit.internal.action	2	100% 22/22	100% 8/8	2
com.pdfunit.internal.guard	2	100% 165/165	90% 29/32	0
com.pdfunit.internal.matcher.document	175	100% 2612/2612	99% 867/868	2,138
com.pdfunit.internal.matcher.page	48	100% 904/904	100% 270/270	2,067
com.pdfunit.internal.util	19	100% 753/753	100% 148/148	2,134
com.pdfunit.messages	276	100% 1159/1159	100% 14/14	1,034
com.pdfunit.rules	9	100% 523/523	99% 126/127	2,372
com.pdfunit.util	4	100% 228/228	100% 41/41	2,645
com.pdfunit.validators	75	100% 2724/2724	99% 403/404	1,304
com.pdfunit.xml	5	100% 85/85	100% 12/12	1,852

Report generated by Cobertura 2.1.1 on 6/4/16 1:14 AM.

100% code coverage (lines)



PDFUnit is well documented

- User manual in English and German
- User manuals 150+ pages
- Same documentation online as HTML
- Many index terms as links, also in PDF
- Javadoc as HTML online and as ZIP for download
- XML–Schema documentation as HTML



PDFUnit at Runtime

- Continuously watching incoming PDF:
 - ZUGFeRD data are as expected
 - No JavaScript in PDF documents
 - Signature exists
 - ...

Example 601

Example 602

Example 603

