

Java pdfbox extract table from pdf

Continue

I would like to find out how to extract from this pdf(ex. image) For example, I want to extract only the values in the column "TENSIONE[V]" and if it encounters a blank cell I enter the letter "X" in the output. How could I do? The code I used is this: PDDocument p=PDDocument.load(new File("a.pdf")); PDFTextStripper t=new PDFTextStripper(); System.out.println(t.getText(p)); and I get this output: 4 How easy would our lives be if there was a way to automate PDF content validation? Ever heard of a Java tool that makes our work easier by extracting the content of a PDF? If you are looking for such a tool, then the Apache PDFBox is what you have been searching for. What Is PDFBox? The Apache PDFBox library is an open-source Java tool for working with PDF documents. It allows us to create new PDF documents, update existing documents like adding styles, hyperlinks, etc., and extract content from documents. Let's get into the details on how to do that! Read Content From a PDF Half of the problem is solved when you extract the text from the PDF. The following code does that for you. Class PDFTextStripper takes a PDF document and strips out all of the text in a document. This ignores all formatting in the document. tStripper = new PDFTextStripper(); tStripper.setStartPage(1); tStripper.setEndPage(3); PDDocument document = PDDocument.load(new File("name.pdf")); document.getClass(); if (!document.isEncrypted()) { pdfFileInText = tStripper.getText(document); lines = pdfFileInText.split("\\r\\n"); for (String line : lines) { System.out.println(line); content += line; } } System.out.println(content.trim()); Obtain All Hyperlinks From a Page in a PDF The second important thing is to validate the PDF by checking the hyperlinks. The following code provides you with the hyperlinks in the document. The getAnnotations() method of PDPage class gives you the list of annotations used in the document. Secondly, fetch the items that are part of the type PDActionURI. This provides a list of URLs used in the document or in a page. PDDocument document = PDDocument.load(new File("name.pdf")); document.getClass(); PDPage pdfpage = document.getPage(1); annotations = pdfpage.getAnnotations(); for (int j = 0; j < annotations.size(); j++) { PDAnnotation annot = annotations.get(j); if (annot instanceof PDAnnotationLink) { PDAnnotationLink link = (PDAnnotationLink) annot; PDAction action = link.getAction(); if (action instanceof PDActionURI) { PDActionURI uri = (PDActionURI) action; urls += uri.getURI(); } } } Extract All Images From a PDF In addition to text and hyperlinks, PDFBox provides the provision to extract images from a document. getResources() method of PDPage class gives you the list of all resource objects (like images). PDDocument document = PDDocument.load(new File("name.pdf")); PDPage pdfpage = document.getPage(1); int i = 1; PDResources pdResources = pdfpage.getResources(); for (COSName c : pdResources.getXObjectNames()) { PDXObject o = pdResources.getXObject(c); if (o instanceof org.apache.pdfbox.pdfmodel.graphics.image.PDImageXObject) { File file = new File(i + ".png"); i++; ImageIO.write((org.apache.pdfbox.pdfmodel.graphics.image.PDImageXObject) o.getImage(), "png", file); } } Get Words That Are Hyperlinked in a PDF Getting text, hyperlinks, and images from a PDF are fairly straightforward tasks. Next, we go a little further to see how to extract hyperlinked words. As we see in the second section, for all 'link' annotations, and for every link, crop the text area by using PDRectangle class. The below code gives you the list of words that are hyperlinked in a document. PDDocument document = PDDocument.load(new File("name.pdf")); int pageNum=0; for(PDPage page : doc.getPages()) { pageNum++; PDFTextStripperByArea stripper = new PDFTextStripperByArea(); List annotations = page.getAnnotations(); //first setup text extraction regions for(int j=0; j 5) return false; value /= 256; } return true; } // // PDFGraphicsStreamEngine overrides // @Override public void appendRectangle(Point2D p0, Point2D p1, Point2D p2, Point2D p3) throws IOException { path.add(new Rectangle(p0, p1, p2, p3)); } @Override public void endPath() throws IOException { path.clear(); } @Override public void strokePath() throws IOException { path.clear(); } @Override public void fillPath(int windingRule) throws IOException { processPath(); } @Override public void fillAndStrokePath(int windingRule) throws IOException { processPath(); } @Override public void drawImage(PDImage pdImage) throws IOException { } @Override public void clip(int windingRule) throws IOException { } @Override public void moveTo(float x, float y) throws IOException { } @Override public void lineTo(float x, float y) throws IOException { } @Override public void curveTo(float x1, float y1, float x2, float y2, float x3, float y3) throws IOException { } @Override public Point2D getCurrentPoint() throws IOException { return null; } @Override public void closePath() throws IOException { } @Override public void shadingFill(COSName shadingName) throws IOException { } // // inner classes // class Interval implements Comparable { final float from; final float to; Interval(float... values) { Arrays.sort(values); this.from = values[0]; this.to = values[values.length - 1]; } Interval(double... values) { Arrays.sort(values); this.from = (float) values[0]; this.to = (float) values[values.length - 1]; } boolean combinableWith(Interval other) { if (this.from > other.from) return other.combinableWith(this); if (this.to < other.from) return false; float intersectionLength = Math.min(this.to, other.to) - other.from; float thisLength = this.to - this.from; float otherLength = other.to - other.from; return (intersectionLength >= thisLength * .9f) || (intersectionLength >= otherLength * .9f); } Interval combineWith(Interval other) { return new Interval(this.from, this.to, other.from, other.to); } @Override public int compareTo(Interval o) { return this.from == o.from ? Float.compare(this.to, o.to) : Float.compare(this.from, o.from); } @Override public String toString() { return String.format("[%3.2f, %3.2f]", from, to); } } interface PathElement { class Rectangle implements PathElement { final Point2D p0, p1, p2, p3; Rectangle(Point2D p0, Point2D p1, Point2D p2, Point2D p3) { this.p0 = p0; this.p1 = p1; this.p2 = p2; this.p3 = p3; } } // members // final List path = new ArrayList(); final List horizontalLines = new ArrayList(); final List verticalLines = new ArrayList(); final Logger logger = LoggerFactory.getLogger(PdfBoxFinder.class); } (PdfBoxFinder.java) Example use You can use the PdfBoxFinder like this to extract text from the table cells of the sample document located at FILE_PATH: try (PDDocument document = PDDocument.load(FILE_PATH)) { for (PDPage page : document.getDocumentCatalog().getPages()) { PdfBoxFinder boxFinder = new PdfBoxFinder(page); boxFinder.processPage(page); PDFTextStripperByArea stripperByArea = new PDFTextStripperByArea(); for (Map.Entry entry : boxFinder.getRegions().entrySet()) { stripperByArea.addRegion(entry.getKey(), entry.getValue()); } stripperByArea.extractRegions(page); List names = stripperByArea.getRegions(); names.sort(null); for (String name : names) { System.out.printf("[%s] %s", name, stripperByArea.getTextForRegion(name)); } } } (ExtractBoxedText test testExtractBoxedTexts) The start of the output: [A1] Nr. crt. [A2] Nume și prenume [A3] Titlul lucrării [A4] Coordonator științific [A5] Ora [B1] 1. [B2] SFETCU I. JESSICA- LARISA [B3] Analiza fluxurilor de date twitter [B4] Conf. univ. dr. Frîncu Marc Eduard [B5] 8:00 [C1] 2. [C2] TARBA V. IONUT- ADRIAN [C3] Test me - rest api folosind java și play framework [C4] Conf.univ.dr. Fortiș Teodor Florin [C5] 8:12 The first page of the document:

i
a
.
u