1、介绍

Apache PDFbox是一个开源的、基于Java的、支持PDF文档生成的工具库，它可以用于创建新的PDF文档，修改现有的PDF文档，还可以从PDF文档中提取所需的内容。Apache PDFBox还包含了数个命令行工具。

Apache PDFbox于2016年4月26日发布了最新的2.0.1版。

备注：本文代码均是基于2.0及以上版本编写。

官网地址：<https://pdfbox.apache.org/index.html>

PDFBox 2.0.1 API在线文档：<https://pdfbox.apache.org/docs/2.0.1/javadocs/>

2、特征

Apache PDFBox主要有以下特征：

PDF读取、创建、打印、转换、验证、合并分割等特征。

3、开发实战

3.1、场景说明

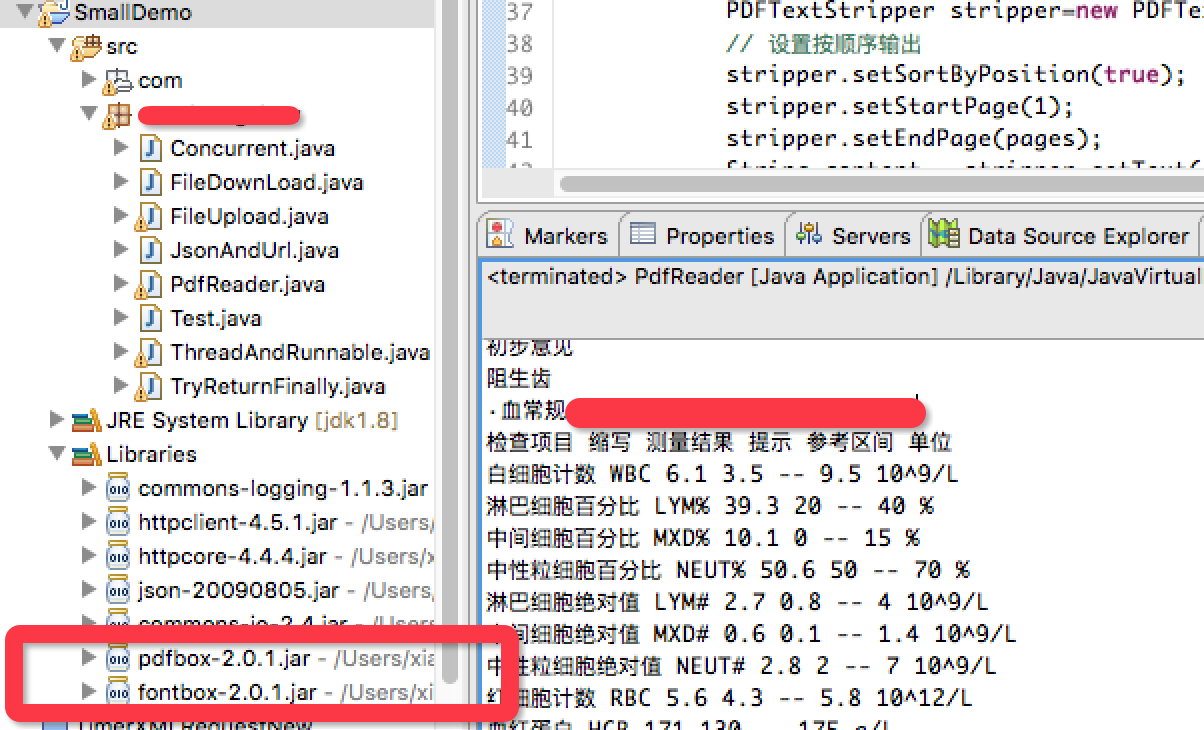
* 1、读取PDF文本内容，样例中为读取体检报告文本内容。
* 2、提取PDF文档中的图片。这里仅仅实现将PDF中的图片另存为一个单独的PDF，至于需要直接输出图片文件（暂时没有实现），大家可以参考我的代码加以拓展，主要就是处理PDImageXObject对象。

3.2、所需jar包

pdfbox-2.0.1.jar[下载地址](http://mirrors.hust.edu.cn/apache/pdfbox/2.0.1/pdfbox-2.0.1.jar)

fontbox-2.0.1.jar[下载地址](http://mirrors.hust.edu.cn/apache/pdfbox/2.0.1/fontbox-2.0.1.jar)

将上述两jar包添加到工程库中，如下：



3.3、文本内容提取

3.3.1、文本内容提取

创建PdfReader类，编写下述功能函数。

package com.loongshaw;

import java.io.File;

import java.io.FileInputStream;

import java.io.InputStream;

import org.apache.pdfbox.io.RandomAccessBuffer;

import org.apache.pdfbox.pdfparser.PDFParser;

import org.apache.pdfbox.pdmodel.PDDocument;

import org.apache.pdfbox.text.PDFTextStripper;

public class PdfReader {

public static void main(String[] args){

File pdfFile = new File("/Users/dddd/Downloads/0571888890423433356rrrr\_182-93201510313223336-2.pdf");

PDDocument document = null;

try

{

// 方式一：

/\*\*

InputStream input = null;

input = new FileInputStream( pdfFile );

//加载 pdf 文档

PDFParser parser = new PDFParser(new RandomAccessBuffer(input));

parser.parse();

document = parser.getPDDocument();

\*\*/

// 方式二：

document=PDDocument.load(pdfFile);

// 获取页码

int pages = document.getNumberOfPages();

// 读文本内容

PDFTextStripper stripper=new PDFTextStripper();

// 设置按顺序输出

stripper.setSortByPosition(true);

stripper.setStartPage(1);

stripper.setEndPage(pages);

String content = stripper.getText(document);

System.out.println(content);

}

catch(Exception e)

{

System.out.println(e);

}

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52

3.3.2、过程说明

PDF文件加载有两种方式，无明显差异，方式二代码较简洁：

// 方式一：

InputStream input = null;

input = new FileInputStream( pdfFile );

//加载 pdf 文档

PDFParser parser = new PDFParser(new RandomAccessBuffer(input));

parser.parse();

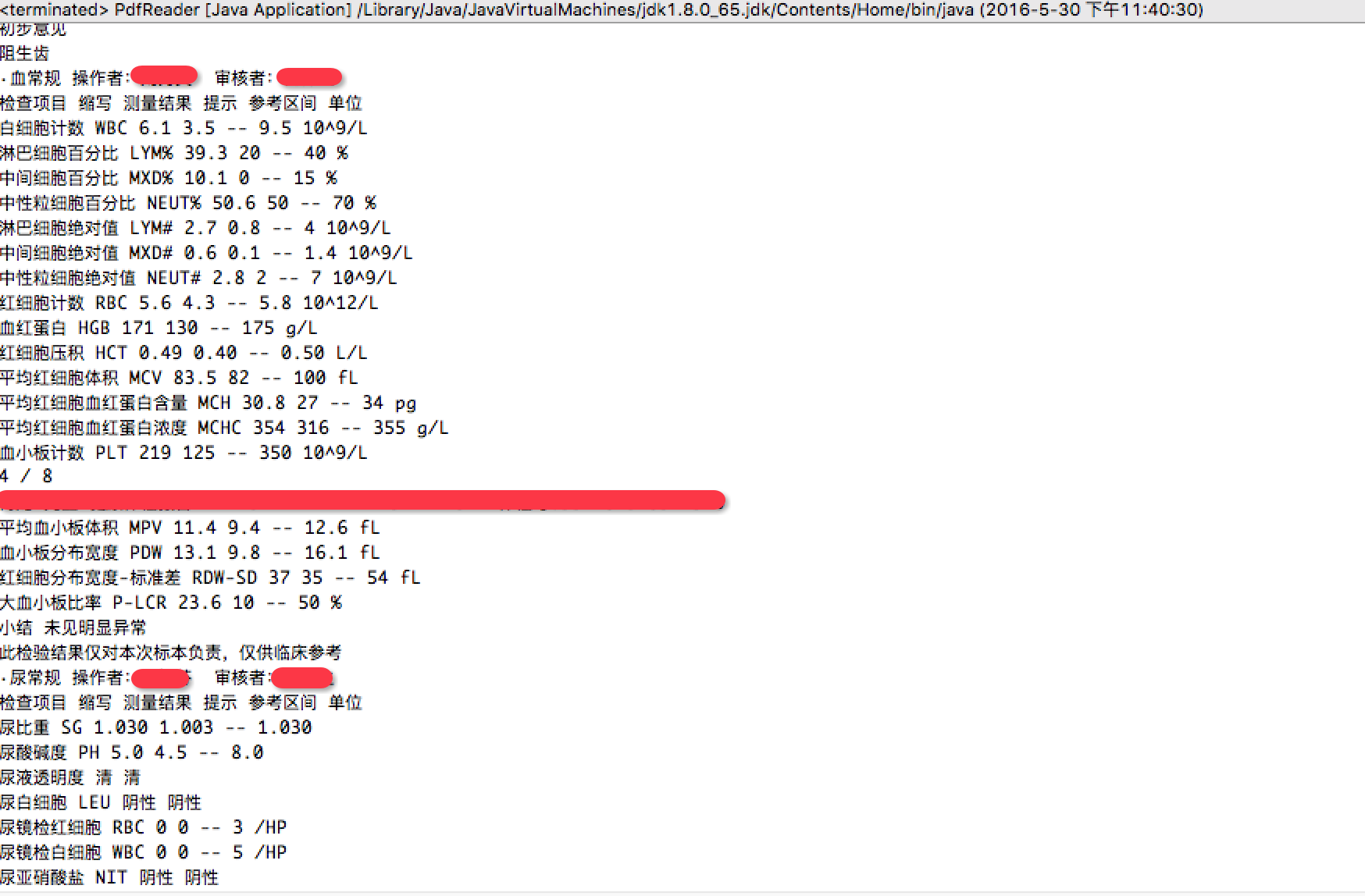
document = parser.getPDDocument();

// 方式二：

document=PDDocument.load(pdfFile);

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9

3.3.3、执行结果



3.4、图片提取（2016-12-02添加）

3.3.1、图片提取

public static void readImage(){

// 待解析PDF

File pdfFile = new File("/Users/xiaolong/Downloads/test.pdf");

// 空白PDF

File pdfFile\_out = new File("/Users/xiaolong/Downloads/testout.pdf");

PDDocument document = null;

PDDocument document\_out = null;

try {

document = PDDocument.load(pdfFile);

document\_out = PDDocument.load(pdfFile\_out);

} catch (IOException e) {

e.printStackTrace();

}

int pages\_size = document.getNumberOfPages();

System.out.println("getAllPages==============="+pages\_size);

int j=0;

for(int i=0;i<pages\_size;i++) {

PDPage page = document.getPage(i);

PDPage page1 = document\_out.getPage(0);

PDResources resources = page.getResources();

Iterable xobjects = resources.getXObjectNames();

if (xobjects != null) {

Iterator imageIter = xobjects.iterator();

while (imageIter.hasNext()) {

COSName key = (COSName) imageIter.next();

if(resources.isImageXObject(key)){

try {

PDImageXObject image = (PDImageXObject) resources.getXObject(key);

// 方式一：将PDF文档中的图片 分别存到一个空白PDF中。

PDPageContentStream contentStream = new PDPageContentStream(document\_out,page1,AppendMode.APPEND,true);

float scale = 1f;

contentStream.drawImage(image, 20,20,image.getWidth()\*scale,image.getHeight()\*scale);

contentStream.close();

document\_out.save("/Users/xiaolong/Downloads/123"+j+".pdf");

System.out.println(image.getSuffix() + ","+image.getHeight() +"," + image.getWidth());

/\*\*

// 方式二：将PDF文档中的图片 分别另存为图片。

File file = new File("/Users/xiaolong/Downloads/123"+j+".png");

FileOutputStream out = new FileOutputStream(file);

InputStream input = image.createInputStream();

int byteCount = 0;

byte[] bytes = new byte[1024];

while ((byteCount = input.read(bytes)) > 0)

{

out.write(bytes,0,byteCount);

}

out.close();

input.close();

\*\*/

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

//image count

j++;

}

}

}

}

System.out.println(j);

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 59
* 60
* 61
* 62
* 63
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73
* 74
* 75
* 76
* 77

3.4.2、过程说明

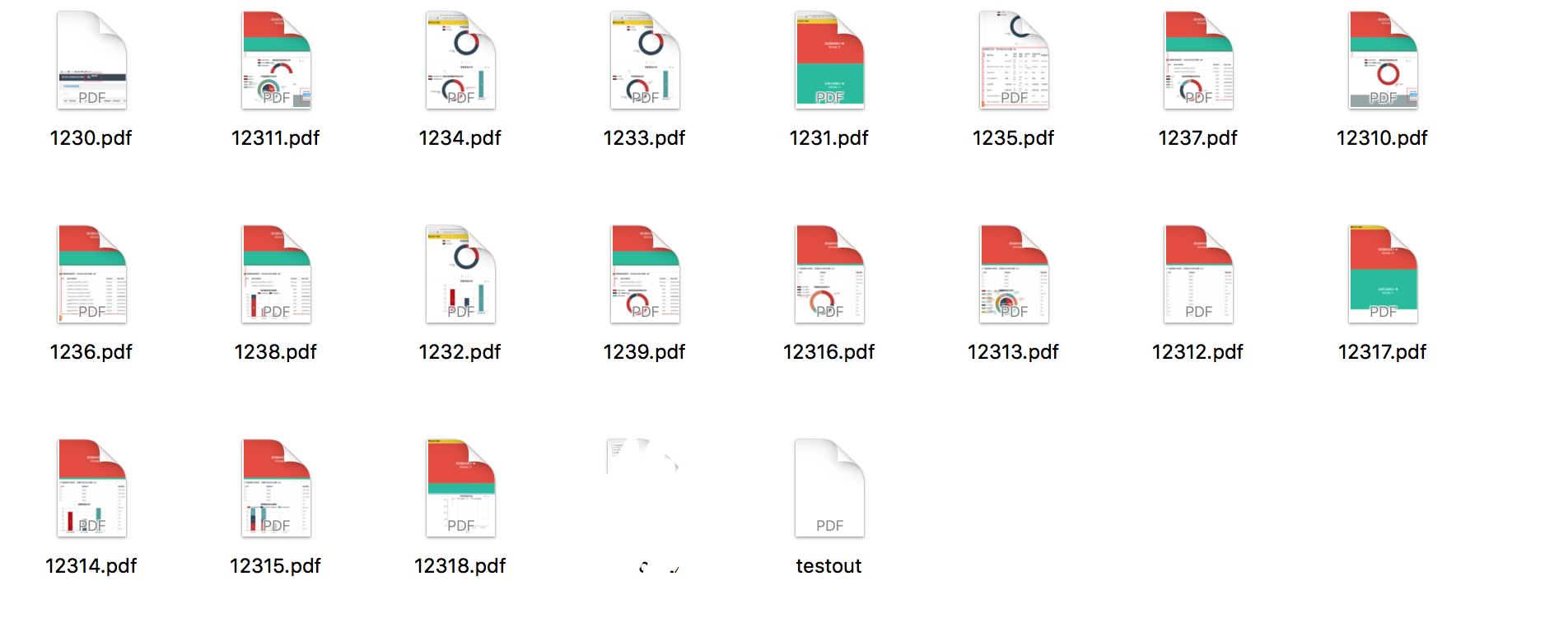
此方法可以取出源PDF中图片对象PDImageXObject，然后可以对该对象进行相关处理，本代码实现了将提取出来的每一个图片对象，插入到一个空白的PDF文档中。

有一点需要说明，以上代码注释部分本意是想直接生成图片文件，但尝试后发现文件异常。因此大家在这个代码基础上有新的想法可以继续尝试。

3.4.3、执行结果



源PDF文件中包含19张图片



分别生成19个仅包含单独图片的PDF

4、小结

本文仅介绍了利用Apache PDFbox相关开发包读取PDF文本，其他复杂功能暂未涉及，需要大家自己线下探索、尝试。