PDFDig Documentation

version 0.2

Micle Bu

July 05, 2012

Contents

PDFDig Documentation

Introduction

PDFDig is a useful tool to dig content from pdf document, which is based on pdftotext [1] and PDFMiner [2].

Features

- Convert pdf to txt.
- Search in pdf document, working like grep.
- Build table of content(TOC) of pdf document.
- Get pdf metadata.

References

PDFDig Tutorial

This tutorial serves as the quick start for PDFDig.

Prerequisites

Python

PDFDig is written in Python, so you should prepare Python environment first. Both Python 2 and Python 3 are OK.

Download Python from:

- http://www.python.org/getit/

Since PDFDig only provides Command Line Interface(CLI) utilities currently, we strongly recommand Windows users to use Cygwin, a linux-like environment for Windows, as running environment for PDFDig to get full features of PDFDig.

Cygwin

Installing Cygwin is pretty easy and straightforward.

- Download setup.exe.
- Run setup.exe and follow its navigation.
- When setup.exe asks you to **Select Packages**, make sure you have selected **Python Default** and then **python: Python language interpreter**.
- After installation, you may try *python --version* within Cygwin Terminal.

pdftotext

PDFDig does not extract content from PDF documents directly by itself, but use a efficient utility, pdftotext, which is freely available and included by default with many Linux distributions. Xpdf provides a pdftotext port to Windows platform.

Windows users:

- Download xpdf binaries, looks like **xpdfbin-win-3.03.zip**, from: http://www.foolabs.com/xpdf/download.html
- Extract xpdfbin-win-3.03.zip in your favorite directory, take D:\ as an example, you'll get D:\xpdfbin-win-3.03.
- pdftotext.exe locates in D:\xpdfbin-win-3.03\bin32\ or D:\xpdfbin-win-3.03\bin64\

- Choose correct version of pdftotext depending on your system architecture, take 32-bit system as an example, you should use the pdftotext.exe in D:\xpdfbin-win-3.03\bin32\.
- Add pdftotext.exe directory **D:\xpdfbin-win-3.03\bin32** to PATH environment variable to ensure system can find pdftotext.exe.

Unix/Linux users:

pdftotext is available and included by default with many Linux distributions. If pdftotext does not exsit in your system, install poppler-utils package.

Check Test:

You can test pdftotext, just run

```
$ pdftotext -v
Copyright 2005-2011 The Poppler Developers - http://poppler.freedesktop.org
Copyright 1996-2004 Glyph & Cog, LLC
```

Installation

1. Get PDFDig source, the tarball file looks like pdfdig-1.0.tar.bz2.
2. Extract the tarball file.

```
$ tar jxvf pdfdig-1.0.tar.bz2
```

3. Install PDFDig.

```
$ cd pdfdig-1.0
$ sudo python setup.py install
```

After the installation, PDFDig will copy PDFDig library to Python library and install 4 executable utilities in your system.

Utility

PDFDig provides you 4 Command Line Interface(CLI) utilities, helps to process PDF documents and do the text processing in command line, so you may need a terminal in Unix/Linux or run *cmd* in Windows before using these utilities.

Refer to *PDFDig Utility* for details.

PDFDig Utility

PDFDig provides you 4 Command Line Interface(CLI) utilities, helps to process PDF documents and do the text processing in command line, so you may need a terminal in Unix/Linux or run *cmd* in Windows before using these utilities.

1. pdftotext.py

pdftotext.py converts pdf to text. There are tens of similar utilities can do this job, while few of them, including pdftotext, can process line-break, hyphen and extra white spaces appropriately, and some of them render the pdf in physical order, which are unsuitable for multi-column pdf documents.

pdftotext.py uses pdftotext to get the text content of pdf document, then normalizes the text content.

1 pdftotext: http://en.wikipedia.org/wiki/Pdftotext
2 PDFMiner: http://www.unixuser.org/~euske/python/pdfminer/

Usage

```
$ pdftotext.py [options] filename1 ...
```

The **pdftotext.py** script has several options:

-o, --output OUTPUTFILE
Specify the output file. e.g: output.txt

-y, --layout LAYOUT
Maintain the layout of the text. LAYOUT can be:

raw
keep the text in content stream order. This is the default setting.

layout
preserve the original physical layout of the text.

-f, --first-page INT
First page to convert.

-l, --last-page INT
Last page to convert.

-p, --page INT
Specify a page to convert.

-h, --help
Print usage information.

Examples

```
$ pdftotext.py input.pdf
$ pdftotext.py -o output.txt input.pdf
```

2. pdfgrep.py

pdfgrep.py enables you to search and count in pdf files. pdfgrep.py searches in grep-style, which means you can use regular expression in search and get matching lines.

Usage

```
$ pdfgrep.py [options] pattern filename ...
```

The **pdfgrep.py** script has several options:

-o, --output OUTPUTFILE
Specify the output file. e.g: output.txt

-c, --count
Print the number of matches for each input file, instead of normal ouput.

-i, --ignore-case
Ingnore case distinctions.

-f, --file-prefix
Prefix each line of output with input file.

-p, --page-number
Prefix each line of output with page number.

-n, --line-number
Prefix each line of output with 1-based line number within its txt file.

-t, --context NUM
Print at most NUM characters of context around each match. e.g: -t 100

-d, --dictionary PATH
Specify the TOC dictionary directory.

-l, --location
Print the match location within TOC.

-C, --color COLOR
Highlight color. COLOR is red by default, also can be black,red,green,orange,blue,purple,bluegreen or white.

-h, --help
Print usage information.

Examples

```
# search in pdf, support multi-pdf at once
$ pdfgrep.py -in "keword" input1.pdf input2.pdf

# search in directory
$ pdfgrep.py -in "keword" pdf-directory

# search and count
$ pdfgrep.py -c "keword" input.pdf

# support location within TOC
$ pdfgrep.py -nl -o output.txt input.pdf

# change highlight color
$ pdfgrep.py -C blue output.txt input.pdf

# save results in a file with a name of output.txt, highlight doesn't work in this case
$ pdfgrep.py -nl -o output.txt input.pdf
```

Output Formarts

The output of search results are formatted to make it more readable. For example, run

```
$ pdfgrep.py -inf  "brain" input.pdf
```

The output may look like:

```
@F:input.pdf @N: 335   @C:  Longitudinal evaluation of early Alzheimer's disease using brain perfusion...
@F:input.pdf @N: 405   @C:  Near-infrared spectroscopy can detect brain activity...
```

The parameters in outputs with following meaning:

```
@F: prefix output lines with filename.
@N: prefix output lines with line number within pdf text.
@C: indicate the context of matches.
```

3. pdftoc.py

Coming soon...

4. dictviewer.py

Coming soon...

PDFDig Release

PDFDig 0.2 (released 2012-04-19)

This is a update release.

New Features

- Match: Support sentence-based context of matches.
- Match: Support highlight the matches.
- pdfgrep: Support search all files under each directory.
- pdfgrep: Add highlight option.

Fixes

- Text: Fix cross-platform check for pdftotext

PDFDig 0.1 (released 2012-04-10)

This is the initial release.

New Features

- Text: Convert PDF to text using 'pdftotext' and normalize the text. Store text lines as an list object.
- Match: Pattern matching based on Text. Store matches as an list object.
- TOC: Build the Table of Content(TOC) of PDF document, filtering by a provided TOC dictionary.
- pdftotext: a (Command Line Interface) CLI utility based on Text.
- pdfgrep: a CLI utility based on Match.
- pdftoc: a CLI utility based on TOC

Refer to *Introduction* to see the details of New Features.

Indices and tables

- *genindex*
- *modindex*
- *search*

Index

Symbols

P