FdfExtract

Minimalist Package to Extract Notes from FoxIt FDF Documents

PREPARED BY RICHARD J. CORDES
RJ.Cordes@COGSEC.org

PREPARED JUNE 2019

Contents

Executive Summary 1
Introduction 1
  Acronyms and Terms 1
  Requirements 2
  Constraints and Concerns 2
Package Overview 3
  Incoming FDF 3
  Proposed Architecture 5
    FdfRouter 5
    FdfReader 5
    CommentBlock 5
    Comment 5
Future Development 5

Executive Summary

The intent of this project is to develop a highly minimalist package to pull comments from FoxIt FDF documents and output a general-purpose object containing relevant data.

Introduction

Forms Data Format files are plain text files which generally contain forms data. However, they are in general use for most text file exports from PDFs (portable document format) and carry a large amount of unwieldly formatting data even in the case that the relevant material within the FDF is simple text. This package is not intended to be stand-alone and will only serve to extract comment information from FoxIt Notes, not FDF files in general.

Acronyms and Terms

For the purposes of this document, the following acronyms and terms have been clarified:

Term	Definition
DocLib	The document library, where PDF and similar documents are stored
FDF	Forms Data Format files. Intended for holding forms data, but are in general use for text files exported from PDFs
FCE	FoxIt Comment Export. A file of type FDF which is the result of an export of annotations from a FoxIt PDF. Holds formatting and other data as XML.
DTD	Document Type Definition. A definition of a document type as per SGML markup language

	standards. XML and HTML are examples of mark-up languages.
AO	Annotation Object. Defined within FCE. Contains comment text.
SPO	Source Path Object. Defined within FCE. Contains source path of the pdf the FCE was generated from.

Table 1: Terms and Acronyms

Requirements

This is a minimalist package; its only requirement is that it be able to facilitate pulling notes and *some* meta-data from FoxIt Comment Exports (FCEs) which come in the form of FDFs, as previously noted.

Constraints and Concerns

There have been several items which are expected to impact and limit the design of the package. While each has the potential to be remedied by additional features, these limitations are beyond the scope of the project and must be considered.

- Document Meta-Data (author, title, publishing information) not included in the FDF.
- FDF has been found to occasionally include notes which have been deleted. It will include an annotation object, but no inner text. The reason for this has not been found and replication of the bug has not been successful. However, it is known to exist.
- Must be built with the assumption that there was no naming/declaration convention or Document Type Definition (DTD) used by the author of the notes.
- The structure of the incoming FDF is predefined and is not guaranteed to remain unchanged in its format

Package Overview

The package, being minimalist, should not require any notable amount of code and will be written in Golang.

Incoming FDF

While the format of the FCE is not *expected* to change, it is not guaranteed to remain stable nor is it known to have always been the format in use. That being the case, the relevant objects and structure should take this into account.

Current FCEs begins with an initial heading:

```
%FDF-1.2
```

Code Snippet 1: FCE Heading

Beyond the heading, the FCE is composed of objects:

```
2 0 obj

<</C[ 1 1 0.192157]/Rect[ 561.375 355.75 581.375 375.75]/F 28/Subj(Note)/Name/Comment/Popup 3 0 R /M(D:20190616210616-05'00')/CreationDate(D:20190616210525-05'00')/Page 0/RC(<?xml version="1.0"?><body xmlns="http://www.w3.org/1999/xhtml" xmlns:xfa="http://www.xfa.org/schema/xfa-data/1.0/" xfa:APIVersion="Acrobat:11.0.0" xfa:spec="2.0.2"><p dir="ltr"><span style="text-align:left;font-size:13pt;font-style:normal;font-weight:normal;color:#000000;font-family:Arial">$NOTE-TEXT$&#x0A;</span></p><p dir="ltr"><span style="text-align:left;font-size:13pt;font-style:normal;font-weight:normal;color:#000000;font-family:Arial">&#x0A;</span></p><p dir="ltr"><span style="text-align:left;font-size:13pt;font-style:normal;font-weight:normal;color:#000000;font-family:Arial">$NOTE-TEXT$&#x0A;</span></p><p dir="ltr"><span style="text-align:left;font-size:13pt;font-style:normal;font-weight:normal;color:#000000;font-family:Arial">&#x0A;</span></p><p dir="ltr"><span style="text-align:left;font-size:13pt;font-style:normal;font-weight:normal;color:#000000;font-family:Arial">&#x0A;</span></p><p dir="ltr"><span style="text-align:left;font-size:13pt;font-style:normal;font-weight:normal;color:#000000;font-family:Arial">$NOTE-TEXT$</span></p></body>)/T(XYZ)/Contents($NOTE-TEXT$\r\n\r\n$NOTE-TEXT$\r\n\r\n\r\n$NOTE-TEXT$)/Subtype/Text/Type/Annot/Rotate 0/CA 1/NM(9f3bfd4a-81de-4949-96cc-e358afd70507)>>

endobj
```

Code Snippet 2: FCE Annotation Object (Relevant Data Emphasised)

As can be seen from the annotation object above, there is a large amount of formatting data that comes alongside simple note text. The note that this object is associated with only contains 33 characters. This accompanying formatting data helps not just to format the text within the note correctly, but also to format the note's container as well. The note that this object is associated with only contains 33 characters.

Luckily, regardless of how much the annotation object may change in the future, it is highly unlikely for FoxIt to cease using FDF, generally speaking. Therefore, in some form or another, it will likely remain in a keep to XML's DTD format.

From each annotation object, all that needs to be retrieved is the inner comment text and the page number. This isn't too concerning, given that the document stores the former towards the top and the latter in full towards the bottom. In addition, it stores each item in a single line, and delimits objects using the x y obj and endobj tags.

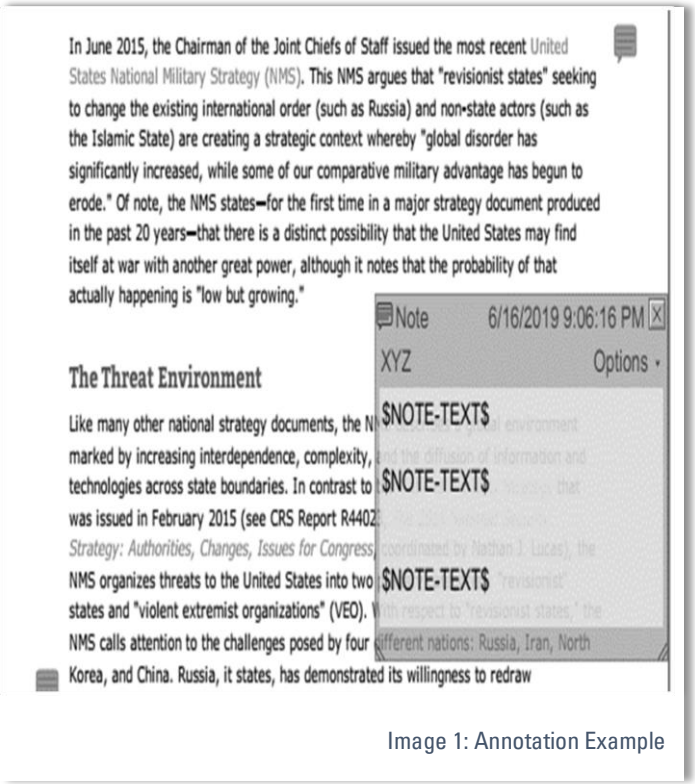

Image 1: Annotation Example

Some light clean-up on the extracted text may be required, but beyond this, there isn't much expected in terms of requirements for text formatting.

There are some other objects mixed in, one of which is the FDF Source Path, which is of some value:

```
1 0 obj

<</FDF<</F(/C/Users/XYZ/Google Drive/DocLib/CRS Insights The 2015 National Military Strategy Background and Questions for Congress - Mcinnis.pdf)/Annots[ 3 0 R  2 0 R  5 0 R  4 0 R  7 0 R  6 0 R  9 0 R  8 0 R ]>>>>

endobj
```

Code Snippet 3: FDF File Header

This Source Path Object (SPO) may or may not be towards the top of the file. More commonly it would seem to be mixed somewhere towards the middle. It contains the file path and name of the PDF the FCE is sourced from. From this SPO comes the only piece of document-level metadata of interest, which is the FilePath of the original Pdf. This is of interest as users may now make use of naming conventions as a means of transmitting additional information via file and folder names.

Proposed Architecture

The proposed structure of the code is straightforward.

FdfRouter

FdfRouter is a function type which should be able to match fdf formats to appropriate FdfReaders.

FdfReader

FdfReader is an interface which should have the method readfdf. Objects which Read FDFs should be able to identify comments and the SPO, parse the data from within and package them into a CommentBlock object for use elsewhere.

** It should also be able to detect an FDF version and choose DTD schemas as necessary.*

CommentBlock

CommentBlock is an object which is expected to contain the document's Source Path and a list of comment objects.

Comment

A comment is an object type which contains text and a page number.

Future Development

In the future, it is expected that this may be expanded to include FDF annotation formats and that the objects will have methods to export to CSV and JSON, but most development should be found in packages which make use of FdfExtract.