Utilizing SAS® and Groovy to combine multiple RTF/PDF reports to one bookmarked PDF document

Xin Weiquan J&J Pharma; Wang John J&J Consumer

Abstract

As part of clinical trial reporting, large numbers of RTF/PDF outputs are created and at the completion of a major milestone in a study, we are often required by medical writers, clients, and regulatory agencies to combine all reports in a user-friendly file format document for easy delivery and review process. One solution is to combine them to a bookmarked PDF document using Adobe Acrobat Software. However, manually generating a PDF document from multiple SAS output files is a time consuming task.

This paper presents an alternative efficacy method that use Java script for combining the multiple RTF/PDF documents into one bookmarked PDF document. A SAS built-in mechanism dynamic language PROC GROOVY which runs on the Java Virtual Machine (JVM) is used to quickly combine the multiple documents into one cumulative file. We have replaced the manual process by automating the ordering of multiple SAS outputs, and the creation of bookmarks within the PDF document. If needed, this script can be used independently.

Introduction

In Pharmaceuticals/CRO industries, the statistical analyses are presented in multiple SAS outputs (i.e., tables, listings, and graphs (TLGs)).These results are created from multiple SAS programs and are delivered to the medical writers and clients for review. It may be impractical for clients to handle each output file individually, especially in cases where there are a large number of outputs. These single files can be created in different user-friendly formats such as Microsoft® Word, HTML, and Portable Document Format (PDF). It is easier to browse and print outputs that are consolidated into one file, rather than handling each output individually. One solution is through the creation of a single bookmarked PDF file.

The PDF standard allows users to exchange and view the electronic documents across platforms and operating systems regardless of the environment in which they are created. The FDA also prefers that electronic submissions are created in the PDF format. It is easier for them to review and provide comments using the track change option. Meanwhile, having all the comments back in one document will make programmers' job easier and more time-saving.

While, the process of converting outputs is often not streamlined and is implemented as a manual process, which always cost substantial time. Unfortunately, SAS does not provide a function to combine multiple RTF/PDF documents. Also, before the finalization of the study draft listings and tables, they are reviewed frequently. In this article, we will illustrate an efficacy method using SAS® 9.4 and GROOVY to generate a single PDF file with user-defined sequence bookmark which can be very useful for interactive navigation.

Technique & mechanism

The techniques presented following offer a good overview of basic data step programming and basic Groovy script. As there are many methods that can automatically transfer RTF files to PDF files, such as two separate software applications (an ASCII-to-PostScript converter and Adobe Acrobat Distiller). We will only focus on the implementation on the multiple PDF files in one source folder. All files are combined when they are in the common file format of PostScript.
For Groovy script, it is a dynamic language on the Java Virtual Machine (JVM). In SAS system, Proc Groovy will execute Groovy code on the JVM, and it can run statements in external files that are specified. It can parse Groovy statements into Groovy Class objects, and run these objects or make them available to other Groovy statements or Java DATA Step Objects. We can also use Proc Groovy to update the CLASSPATH environment variable with additional CLASSPATH strings or file refs to jar files.

A Flowchart of the main process involved in our method is shown below (figure 1). We demonstrate the approach in three steps.

Step 1: The user manually enters or updates the name sequences of desired SAS output files in a Microsoft® Excel spreadsheet which serve as a table of contents (TOC).

Step 2: Scanning a folder using SAS File I/O functions to identify the PDF files to be combined and creating the JSON file which can be read in the JVM.

Step 3: Using Proc groovy to combine multiple PDF into a single one.

Steps in detail

Step 1

Generally, the TOC file is created based on the DPS which has a dual purpose. First, it is used in the procedure of creating tables, listings, and graphs in order to dynamically name each output file, and generate the text and numbers in the title on each TLG. Second, the TOC can be used to indicate which files are to be included in the final PDF document after all the TLGs are created. The row order of each output file name in the TOC dictates the sequence order of the output file presented in the final PDF document.

The TOC hold several important parameters for each TLG file (Figure 2):

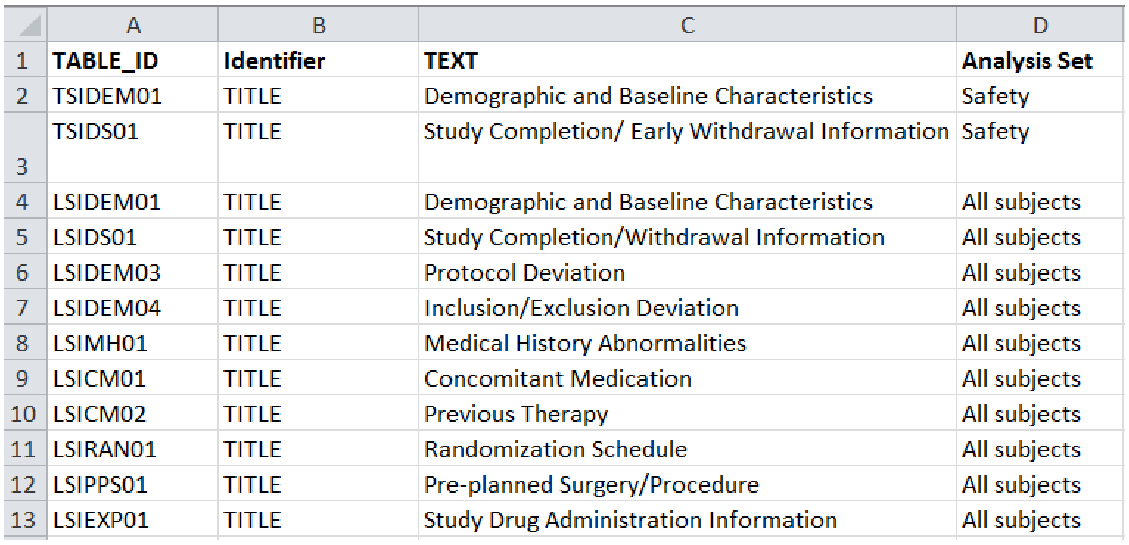
i) A variable as TLG number which identifies the file as a table, listing, or graph.

ii) An identifier which indicate the elements of each TLGs;

iii) The text column which included associated title for each TLG;

iv) The population definition.

All the TOC data is manually entered into a Microsoft® Excel spreadsheet with one row per output file. The TOC spreadsheet contains all the relevant parameters needed to combine all the output files. These parameters are the output file names and the sequence- which are to appear in the overall PDF document. If the user wants to change the order of any output file, the corresponding spreadsheet row order is manually altered. Thus, the user is flexible to change the sequence of files presented in the overall PDF document by manually updating and saving the TOC spreadsheet.



	A	B	C	D
1	**TABLE_ID**	**Identifier**	**TEXT**	**Analysis Set**
2	TSIDEM01	TITLE	Demographic and Baseline Characteristics	Safety
3	TSIDS01	TITLE	Study Completion/ Early Withdrawal Information	Safety
4	LSIDEM01	TITLE	Demographic and Baseline Characteristics	All subjects
5	LSIDS01	TITLE	Study Completion/Withdrawal Information	All subjects
6	LSIDEM03	TITLE	Protocol Deviation	All subjects
7	LSIDEM04	TITLE	Inclusion/Exclusion Deviation	All subjects
8	LSIMH01	TITLE	Medical History Abnormalities	All subjects
9	LSICM01	TITLE	Concomitant Medication	All subjects
10	LSICM02	TITLE	Previous Therapy	All subjects
11	LSIRAN01	TITLE	Randomization Schedule	All subjects
12	LSIPPS01	TITLE	Pre-planned Surgery/Procedure	All subjects
13	LSIEXP01	TITLE	Study Drug Administration Information	All subjects

Figure 2: example of TOC---titles.xls

Step 2

Then we can scan a folder using SAS File I/O functions to identify the PDF files to be combined. And we will combine with TOC file to monitor the complete status of the RTF files. Then we create a JSON file named as toc.json which can be read in the JVM.

```
%*step 1: scan the name of RTF/PDF file in source folder;
x "dir /b &rptdrv\......\PDF\*.PDF  > &rptdrv\......\PDF\pdfname.txt";
filename inrtf " &rptdrv\......\PDF\ pdfname.txt";

data comprtfname0;
       infile inrtf truncover ;
       input  rtfname $100.;
       run;
quit;

data comprtfname;
       set comprtfname0;
       rtfname=strip(upcase(rtfname));
       rtfname=tranwrd(rtfname,'PDF','RTF');
run;

%*step 2: import title from TOC spread sheet into SAS dataset;
proc import dbms=xls out=allrtfxls
```

```
       datafile="&rptdrv\Prod\DPS\titles.xls"
       replace;
run;

data allrtfxls_tit;
       set allrtfxls;
       where identifier='TITLE';
       length rtfname $100.;
       rtfname=strip(upcase(table_id))||'.RTF';
       keep table_id identifier text rtfname;
run;

%*Monitor the complete RTF files in study folder;
proc sql;
       create table compxlsrtf as
       select *
       from allrtfxls_tit
       where rtfname in (select distinct rtfname from comprtfname);
quit;

data toc(keep =TLF_ID TLF_Num TLF_Title);;
       set compxlsrtf;
       length TLF_ID TLF_Num $60 TLF_Title $200;
       TLF_ID=table_ID;
       TLF_Num=table_ID;
       TLF_Title=TEXT;
run;

%*generate the json file for use;
filename jsonfile "&outpath.\...\toc.json" encoding="utf-8";
proc json out=jsonfile pretty ;
    export toc;
run;
quit;
```

Step 3

While using the Proc Groovy, we firstly update the CLASSPATH environment variable with additional CLASSPATH strings or referencing several external jar files (e.g itext, pdfbox,etc).

e.g

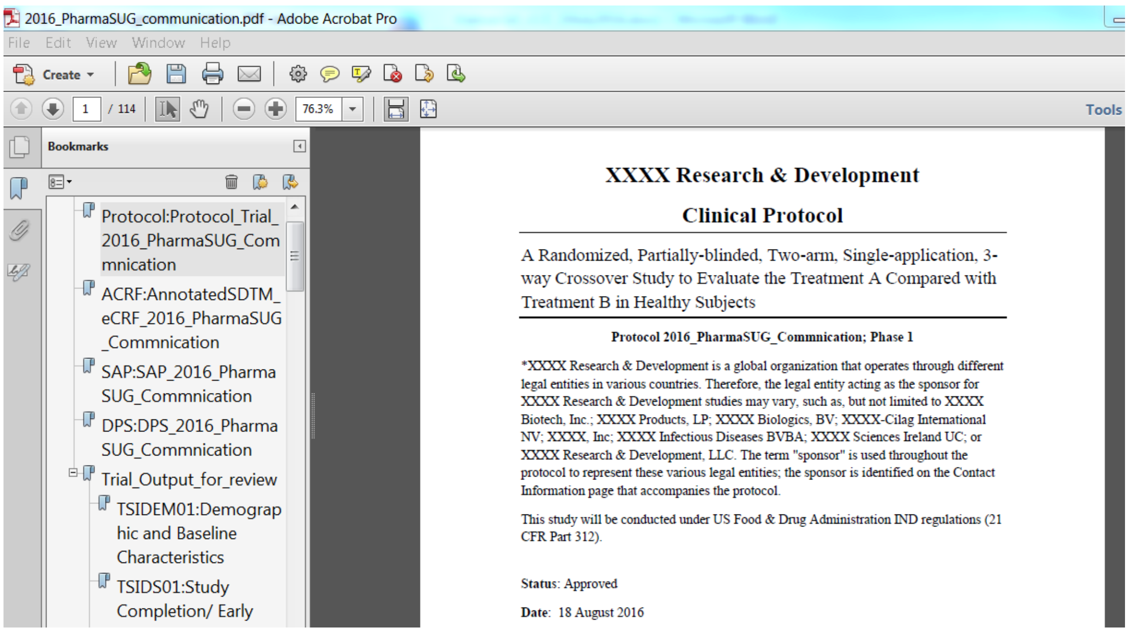
```
add classpath="C:\Temp\groovy-2.4.5\embeddable\groovy-all-2.4.5.jar";
add classpath="&pboxdir.\pdfbox-1.8.10.jar;&pboxdir.\pdfbox-app-
1.8.10.jar;&pboxdir.\fontbox-1.8.10.jar";
……
```

Then we write Groovy script to combine multiple PDF files into a single one and store it in the determinate folder as user defined, e.g., “PDFOUT ”. In this process, the source folder which stores multiple PDF files and the JSON file should be induced.

```
import java.io.IOException
import java.io.File
import java.util.HashMap
import groovy.io.FileType
import groovy.json.JsonSlurper

......

public class MergePDFBookmarks
{
      public static void mergePDFFiles(Map jsonobj, String pathname, String 
pdfoutfullpath)
      {
            def mergePdf = new PDFMergerUtility();

            jsonobj.each{k,v->
                  if (v instanceof List){
                  v.each{it ->
                  mergePdf.addSource(pathname + "/" + it["TLF_ID"] + ".pdf")
                        }
                  }
            }

            // Save and close the combined PDFs.
            println "Merging to PDF File: " + pdfoutfullpath + ".pdf"
            mergePdf.setDestinationFileName(pdfoutfullpath)
            mergePdf.mergeDocuments()
      }
public static void createBookmarks(Map jsonobj, String pathname, String 
pdfoutfullpath)
      {
            // load the combined PDF document
            def pdfDocument = PDDocument.load(pdfoutfullpath)
            def totalPages = 0
            def outline =  new PDDocumentOutline()
            pdfDocument.getDocumentCatalog().setDocumentOutline( outline )
            def pages = pdfDocument.getDocumentCatalog().getAllPages()
            def pagesOutline = new PDOutlineItem()

            pagesOutline.setTitle("Tables")
            outline.appendChild(pagesOutline)

            jsonobj.each{k,v->
            if (v instanceof List){
            v.each{it ->
      def pdfDoci = PDDocument.load(pathname + "/" + it["TLF_ID"] + ".pdf")
            def pdfDocPages = pdfDoci.getNumberOfPages()
            pdfDoci.close()

            totalPages = totalPages + pdfDocPages
            println it["TLF_NUM"] + ": " + it["TLF_TITLE"]
            def page = pages.get( totalPages - pdfDocPages);
            def dest = new PDPageFitWidthDestination()
            dest.setPage(page)

            def bookmark = new PDOutlineItem()
            bookmark.setDestination(dest)
            bookmark.setTitle( it["TLF_NUM"] + ":" + it["TLF_TITLE"] )
            pagesOutline.appendChild( bookmark )
                        }
                  }
            }

            pagesOutline.openNode()
```

```
                outline.openNode()

                // Save and close the combined PDFs.
                pdfDocument.save(pdfoutfullpath)
                pdfDocument.close()
        }

    public static void main( String[] args ) throws Exception
    {
def path   = "……/PDF"
def toc    = "……/toc.json"
def pdfout = "……/2016_PharmaSUG_present.pdf"
def jsonstr = new File(toc).getText('UTF-8').toString()
jsonstr = jsonstr.substring( 1)

def jsonSlurper = new JsonSlurper()
def tocjson = jsonSlurper.parseText(jsonstr)

MergePDFBookmarks.mergePDFFiles(tocjson, path, pdfout)

MergePDFBookmarks.createBookmarks(tocjson, path, pdfout)

println "Combining all of the PDF are Completed!"
          }
}
```

The combined PDF file can be convenient to be reviewed by the interactive navigation through the right bookmark. (Figure 3).


2016_PharmaSUG_communication.pdf - Adobe Acrobat Pro
File Edit View Window Help
Create
1 / 114
76.3%
Tools
Bookmarks

Protocol:Protocol_Trial_2016_PharmaSUG_Commnication
ACRF:AnnotatedSDTM_eCRF_2016_PharmaSUG_Commnication
SAP:SAP_2016_PharmaSUG_Commnication
DPS:DPS_2016_PharmaSUG_Commnication
Trial_Output_for_review
TSIDEM01:Demographic and Baseline Characteristics
TSIDS01:Study Completion/ Early

XXXX Research & Development

Clinical Protocol

A Randomized, Partially-blinded, Two-arm, Single-application, 3-way Crossover Study to Evaluate the Treatment A Compared with Treatment B in Healthy Subjects

Protocol 2016_PharmaSUG_Commnication; Phase 1

*XXXX Research & Development is a global organization that operates through different legal entities in various countries. Therefore, the legal entity acting as the sponsor for XXXX Research & Development studies may vary, such as, but not limited to XXXX Biotech, Inc.; XXXX Products, LP; XXXX Biologics, BV; XXXX-Cilag International NV; XXXX, Inc; XXXX Infectious Diseases BVBA; XXXX Sciences Ireland UC; or XXXX Research & Development, LLC. The term "sponsor" is used throughout the protocol to represent these various legal entities; the sponsor is identified on the Contact Information page that accompanies the protocol.

This study will be conducted under US Food & Drug Administration IND regulations (21 CFR Part 312).

Status: Approved

Date: 18 August 2016

Figure 3: example of combined PDF

Discussion and Conclusion

This paper introduced a method to combine multiple PDF/RTF documents using PROC GROOVY in SAS®, which can substantially save time than traditional process and keep the accurate of the whole combination. The techniques we presented offer a basic Groovy program and are easy to use by programmers at all levels. The Groovy script can also be used independently in the JAVA editor such as ECLIPS. To consider the limit of the cost memory of JVM, we should pay attention to the situation when there are too many PDF files to be combined. We may split multiple PDF file into several parts and combine them together.

DISCLAIMER

The contents of this paper are the work of the authors and do not necessarily represent the opinions, recommendations, or practices of their respective organizations.

Acknowledge

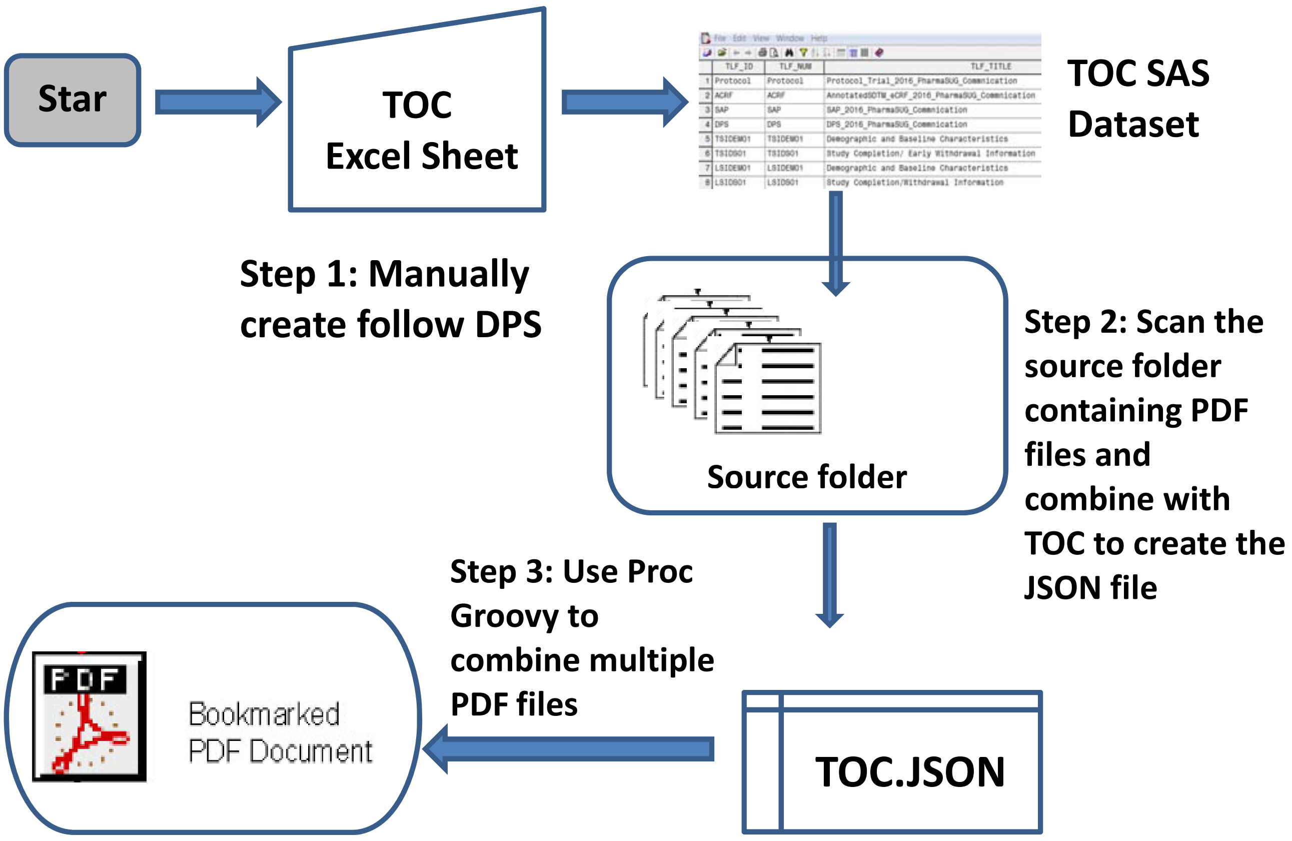
SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute in the USA and other countries.
I would like to thank John. He gave me moral support and guided me in different matters regarding the topic. I am equally grateful to George for helping reviewing the paper to correct many explain and programming error. I am very thankful to Prof…. for his proof reading and comments.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

John Wang, Principle Scientist, JnJ
Weiquan Xin, Pharma programmer, JnJ.

25F, Shinmay Union Square
No.999 South Pudong Road
E-mail:wxin1@its.jnj.com



	TLF_ID	TLF_NUM	TLF_TITLE
1	Protocol	Protocol	Protocol_Trial_2016_PharmaSUG_Commnication
2	ACRF	ACRF	AnnotatedSDTM_eCRF_2016_PharmaSUG_Commnication
3	SAP	SAP	SAP_2016_PharmaSUG_Commnication
4	DPS	DPS	DPS_2016_PharmaSUG_Commnication
5	TSIDEM01	TSIDEM01	Demographic and Baseline Characteristics
6	TSIDS01	TSIDS01	Study Completion/ Early Withdrawal Information
7	LSIDEM01	LSIDEM01	Demographic and Baseline Characteristics
8	LSIDS01	LSIDS01	Study Completion/Withdrawal Information

Figure 1: Flowchart process of combining multiple PDF files