PhUSE US Connect 2019

Paper CT02

SAS is still a winner – with a little help from Java

Praveen Kumar J, Ephicacy, Bangalore, India

ABSTRACT

SAS provides a one-stop-shop solution for completing statistical programming tasks. It is well known for the wide array of functions and features for handling most activities that are typically a part of statistical programming processes. However, one challenging task that necessitates the use of a third-party tool/command is that of combining PDF reports into one combined package for easier delivery. Java, a platform independent programming language, can come to the rescue for the task of combining PDFs into a single file without the use of third-party software, in addition to ensuring platform independence. This paper will demonstrate a SAS program that allows SAS to interface with Java to combine multiple PDFs into a single file.

INTRODUCTION

In our industry, there are many responsibilities that are part of day-to-day activities related to statistical analysis and reporting. In addition to programming and data review, a SAS programmer also needs to perform a few other tasks. One of these tasks is combining multiple report files into a single file. This can be for PDF or RTF documents. This is required during submissions of TFLs (Tables, Figures and Listings) and to create a single report file, a SAS programmer usually resorts to third party tools like acrobat reader which are usually an additional cost to the organization. The other option is to use other processes such as reading an XLSX document. A SAS programmer needs an advanced license from SAS to perform reading operation of XLSX files which is again an additional cost to the company since for programming purposes where data needs to be read from an XLSX file, a SAS programmer usually converts the XLSX file to a CSV file and reads it in using one of the many available methods.

Many third-party tools that can create combined PDF files are created using a front-end application like JAVA®, C# or similar. These third-party tools are to be purchased by the organization and the periodic software upgrades also don't come easy. And, these tools differ from operating system to operating system. For example, a tool that is used in Windows cannot be installed in Linux platform and hence these tools are not platform independent.

SAS provides a solution to all the above-mentioned problems through SAS Private JRE® (Java Runtime Environment). This package enables a user to create a simple solution for the above-mentioned problem. SAS by default, has a third-party tool included called SAS private JRE where a JAVA program can be executed in JRE – Java Runtime Environment. Once a user has identified a problem, they can check if the SAS JRE can solve it for them. If so, then a JAVA program should be created to solve the user requirement and then it should be executed in the SAS environment. The core idea behind JAVA incorporated into SAS is that JAVA is an object-oriented programming language and supports platform independence. Whereas SAS is differentiated by the platform like Windows SAS and Linux SAS, JAVA programs can execute across any platform. So, one JAVA program can be executed in Windows SAS or Linux SAS.

This paper will take the problem of combining multiple PDF files into one PDF file and explain how this can be done by writing a programming in JAVA and executing it in SAS. The paper will start with explaining the requirements and then introduce Java using sample programs that can be run in SAS.

JAVA

JAVA is an Object-oriented programming language and its platform independence is a key concept that helps when Java is used within SAS. Generally, JAVA programs are executed in an environment called JRE – JAVA Runtime Environment. To enable JRE, one needs to know where the JRE resides and in JAVA this path is called CLASSPATH. And each program goes through two phases - a compilation phase followed by an execution phase. During compilation, it generates a CLASS file. The CLASS file is used as an executable file. It is the CLASS file that allows JAVA to be platform independent. CLASS files have values in terms of mnemonic codes which is machine language. This allows any machine to read the values hence allowing platform independence.

ITEXTPDF

The main package which is required in JAVA is "**itextpdf**", also called as the JAR file. This is an open source file available on the web. For PDF related applications, this package needs to be imported. This "**itextpdf**" package controls creation of PDF files and this enables a few properties like adding bookmarks, altering page number and inserting images into a PDF file. JAVA uses import statement to import packages and to use its properties.

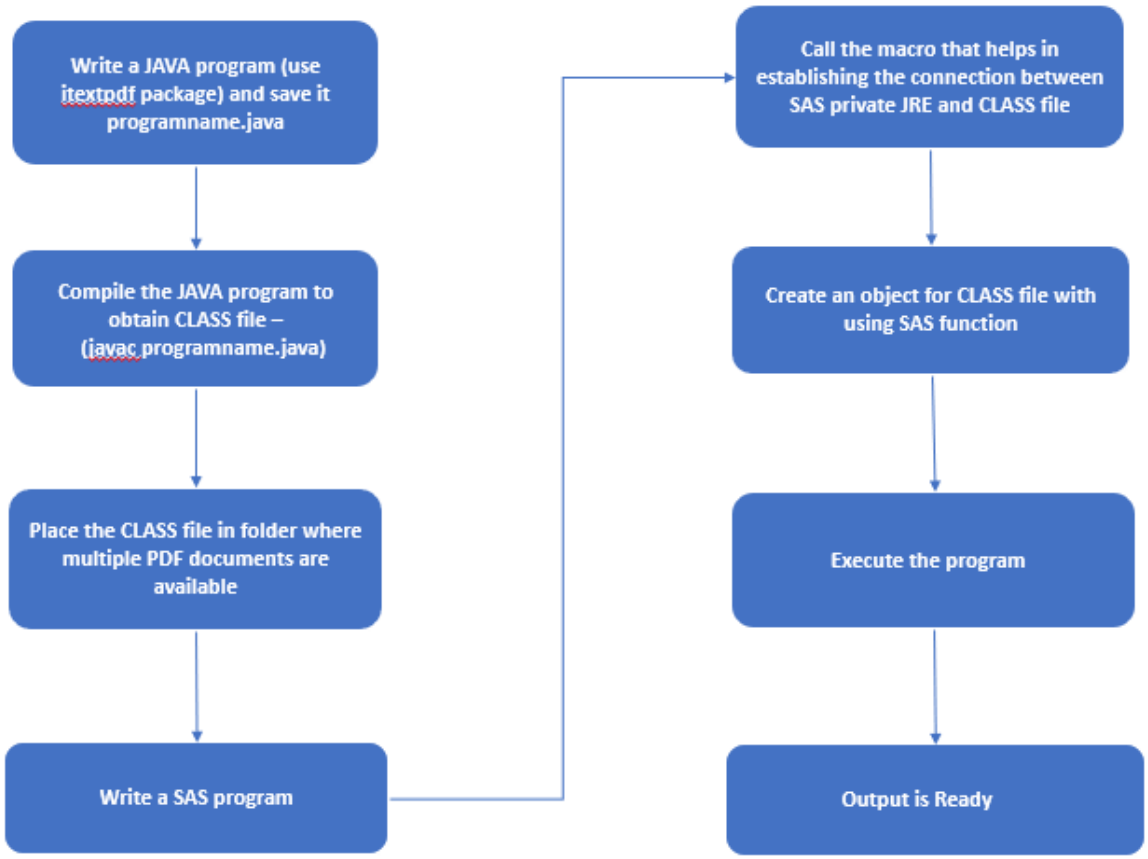
SAS PRIVATE JRE

SAS Private JRE is a third-party tool that is readily available in SAS software as a default package. There is no need of explicit installation of this package. In general, JAVA Runtime Environment is where JAVA program gets executed. SAS provide a place to run these JAVA programs under the control of SAS. As mentioned earlier, during the compilation phase, a CLASS file is generated, and this CLASS file can be used across any SAS (Windows or Linux).

BUILDING THE MACRO

The high-level flowchart of proposed macro is shown in figure 1. Using JAVA front-end programming technology, a user creates a requirement to combine multiple PDFs into a single PDF file. Once the JAVA program is ready, it is then compiled in JAVA environment to obtain the JAVA CLASS file. The user writes a SAS code that calls the CLASS file. During this process, a SAS program will need to have the path where the JAVA CLASS file resides.

Figure 1



SAS PRIVATE JRE AND IT'S MACRO

As discussed above, a JAVA program execution requires the path where JAVA JRE resides. This path is known as CLASSPATH. SAS Private JRE is one kind of CLASSPATH. To enable this CLASSPATH to execute the CLASS file in SAS, SAS by default provides three set of macros available in SAS Support area as detailed below:

%init_classpath_update; → Initialize the CLASSPATH – obtain existing CLASSPATH

%add_to_classpath(<<Classpath>>); → Change of CLASSPATH – Assign the path were CLASS file is residing

%reset_classpath; → Rollbacks to previous CLASSPATH

CODE SNIPPET

- Create a JAVA Program
- Obtain the JAVA CLASS file
- Call SAS macro that generate the link between SAS Private JRE environment and JAVA CLASS file
- Call the CLASS file and execute the program.

JAVA PROGRAM TO COMBINE TWO PDF INTO SINGLE PDF

// All JAVA requires certain packages which comes by default or even one can use an explicit package.

To do a PDF related process there are different packages available online. here the JAVA program uses itextpdf package that one can download online. Please refer to REFERENCES for download link of itextpdf package.

** Import options in JAVA bring packages (JAR/CLASS) and program uses its properties to enhance an application. **;

** com.itextpdf package provides pdf properties to read, write, save, edit, copy and others. **;

** java.io package comes default with JAVA, one can see it in JAVA home directory. here as our task is to read a PDF file and process them, JAVA's default package of io.File is used to take control of File and its properties **;

```
import com.itextpdf.text.Document;
import com.itextpdf.text.DocumentException;
import com.itextpdf.text.pdf.PdfCopy;
import com.itextpdf.text.pdf.PdfReader;
import com.itextpdf.text.pdf.PdfSmartCopy;

import java.io.File;
import java.io.FileOutputStream;
import java.io.FilenameFilter;
import java.io.IOException;
import java.util.List;

** All JAVA programs must have the same name as Class name
   mergpdf, As JAVA is case sensetive language, the programmer
   should be careful about cases. **;

class mergpdf
 {
** mergePDF is function created inside CLASS,
This function uses are,
1. Identify the current folder were user want to concatenate
    the PDF files.
2. Identify .PDf files and creates a list that has names of
    app PDF files
3. Merge or append or combines all file to a single file
**;
```

```
public static void mergePDF(String directory, String targetFile) throws
DocumentException, IOException
 {
File dir = new File(directory);
File[] filesToMerge = dir.listFiles(new FilenameFilter()
 {
public boolean accept(File file, String fileName)
 {
  //System.out.println(fileName);
  return fileName.endsWith(".pdf");
  }
  });
Document document = new Document();
FileOutputStream outputStream = new
     FileOutputStream(directory+"\\"+targetFile);
                    PdfCopy copy = new PdfSmartCopy(document,
          outputStream);
document.open();

for (File inFile : filesToMerge)
{
 //System.out.println(inFile.getCanonicalPath());
 PdfReader reader = new PdfReader(inFile.getCanonicalPath());
 copy.addDocument(reader);
 reader.close();
}
document.close();
}

** This function combinepdf() is what called in our SAS
    program - This function type is STRING and this produces
    the file output. As the above function mergePDF is called
    here as mergePDF()

   System.getProperty("user.dir"): is used to identify the
    current location of file residing .CLASS file or .JAVA      file. In our
context were all PDF documents are available

   In mergePDF function a value of "mergedfile.pdf" is
    provided were all PDF combines in into one single PDF
    file as "mergedpdf.pdf". **;

public static String combinepdf()throws Exception
{
String cwd = System.getProperty("user.dir");
mergePDF(cwd,"mergedFile.pdf");
return "Hello User: JAVA program executed in SAS Window";
}

** The below function is the heart of JAVA, if a user is executing a JAVA program
in a JAVA environment one needs to have this function written and call the above
functions. **;

public static void main(String[] args) throws Exception
{
 //System.out.println("Current working directory : " + cwd);
 combinepdf();
}   }
```

OVERVIEW OF SAS CODE:

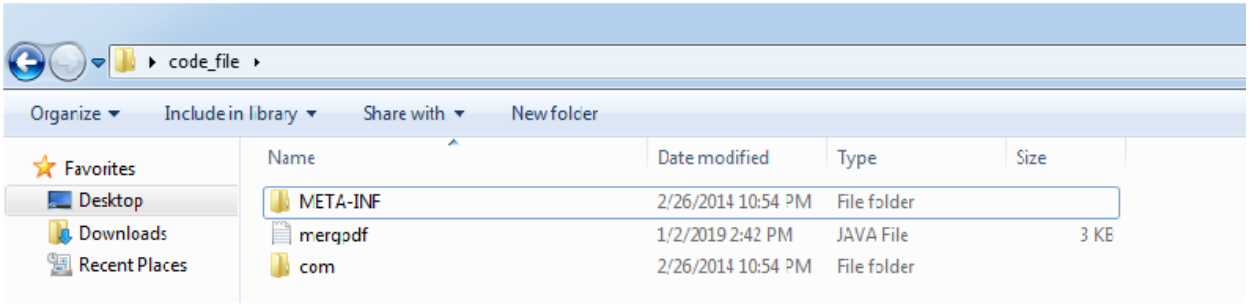
```
%init_classpath_update;


%add_to_classpath(….code_file);


   DATA out;
     length output $200;

   /**
       Syntax to assign an object to CLASS created

       DECLARE JAVAOBJ <<Object Name>> ("<< CLASS FILENAME >>");

       DECLARE keyword initiate a JAVA object for the CLASS Created MERGPDF

       In JAVA Program name MERGPDF is defined as CLASS name

       DECLARE JAVA OBJECT and JAVA Object name is "DF" for CLASS file name MERGPDF
   **/


    DECLARE JavaObj df("mergpdf");


   /**

       SYNTAX to access the properties or function of CLASS file

       <<Object Name>>.callxxxmethod("<<CLASS Properties>>", <<output variable>>);

       Xxx → STATICINT or STATICSTRING or INT or STRING based upon function datatype

       function created in JAVA is string function, so STATICSTRING method is called
   in

       below SAS program

   **/

      df.callStaticStringMethod("combinepdf",output);
      put output=;

   RUN;

   %reset_classpath;
```

THE ACTUAL STEPS TO CREATE AND VIEW THE OUTPUT ARE ILLUSTRATED BELOW IN FIGURES 2 THROUGH 7.

Step 1:

Create a folder and create a JAVA Program (Mergpdf.java).
Com is the package that contains itextpdf properties.

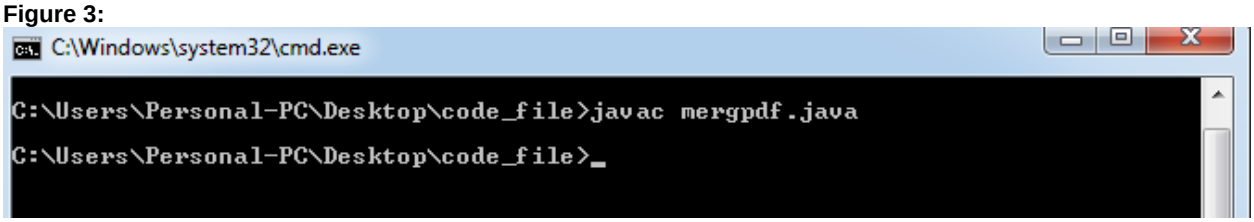
Figure 2:



Step 2:

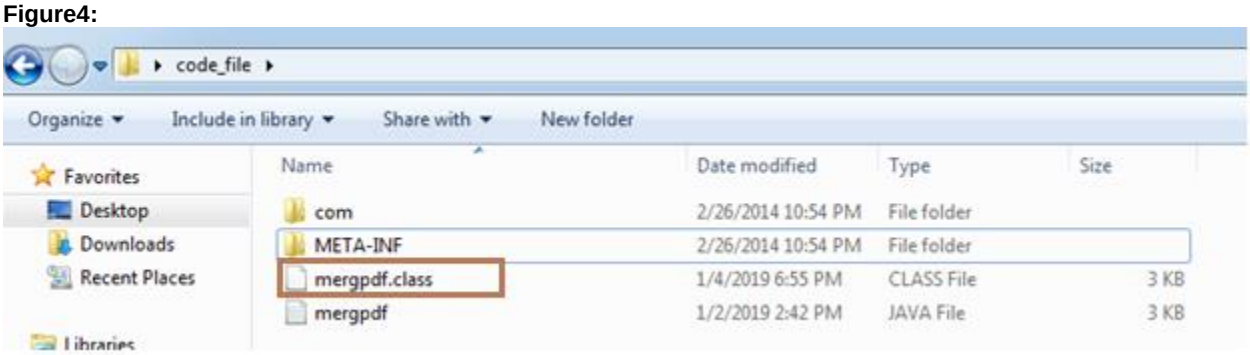
Compile the JAVA program to get the CLASS file.
Navigate to the CLASSPATH and compile using keyword JAVAC mergpdf.java

Figure 3:



After compilation a user could see an additional .CLASS file getting created in the same folder as per **Figure4**.

Figure4:

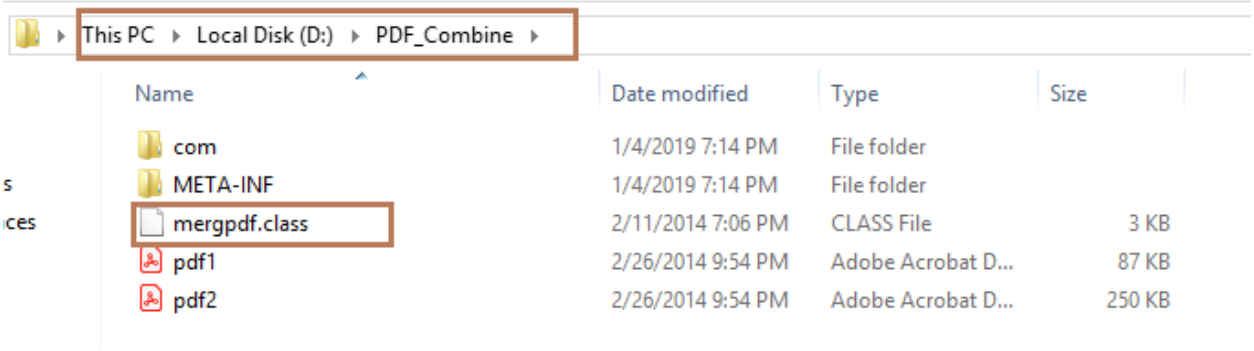


Step 3:
Copy the CLASS (mergpdf.class) file and paste in location were user has PDF files to combine (folder name: PDF_Comfine).
While copying a CLASS file, user should ensure that corresponding package which are used while JAVA programming is also copied along with CLASS file and placed in the required folder.

At that point of time, folder PDF_Combine must have following files,

Mergpdf.class - .CLASS file
COM and META-INF – JAVA Package (ITEXTPDF) which was used while JAVA programming
PDF files- Files that should be combined as one document.

Figure 5:



Step 4 and Final result:

Specify the path in %add_to_classpath macro like %add_to_classpath(D:\PDF_Combine).
Please see above SAS code place for the change of path information.

After assigning the path, execute the program to get your final PDF (Mergerp.pdf) as described in JAVA program file.

Figure 6:

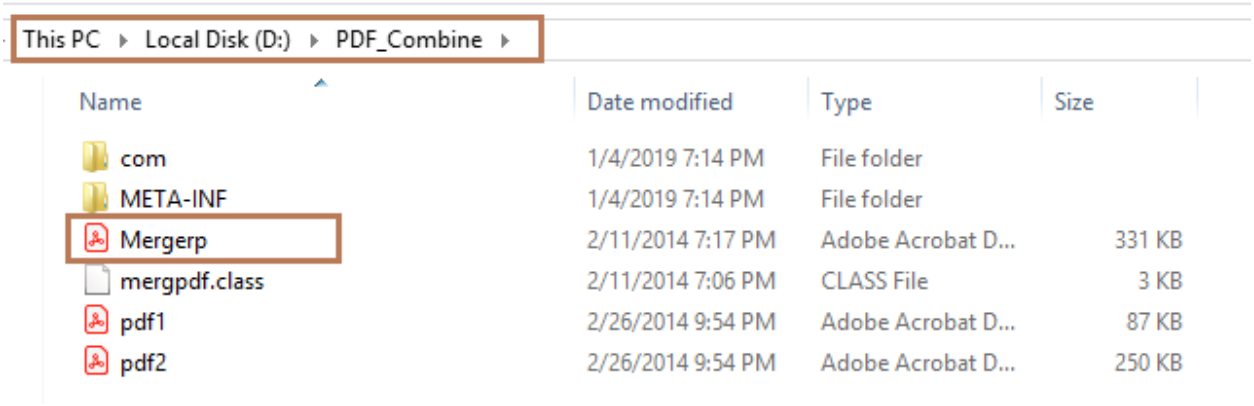
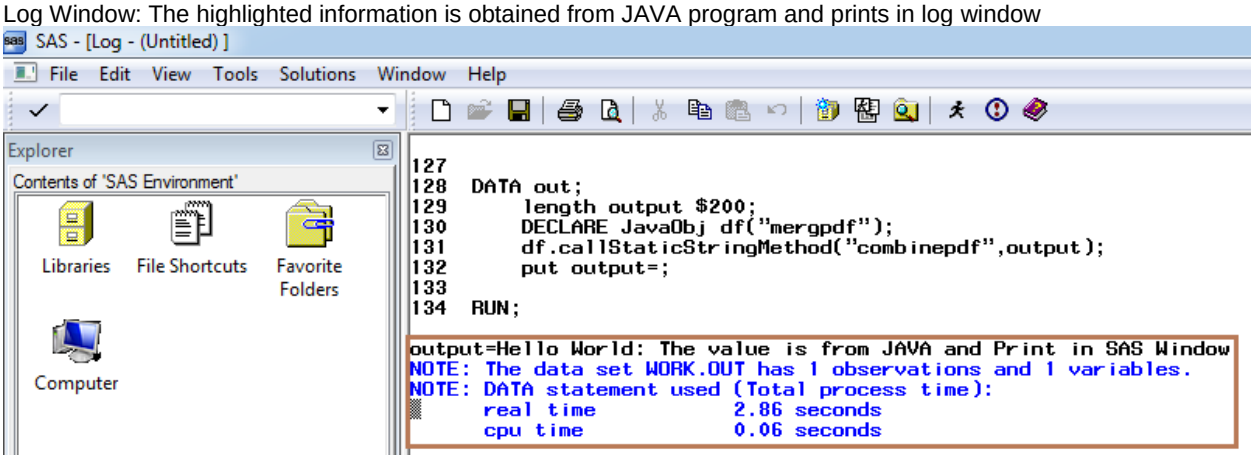


Figure 7:

Log Window: The highlighted information is obtained from JAVA program and prints in log window



CONCLUSION

There are plenty of programming technologies that can help make our work easier. JAVA is one type of technology that has solved many problems that users have faced. JAVA technology can help in creating many kinds of applications. In our day-to-day activities, we also come across situations which take our time away from programming. A common problem faced by SAS programmers is for a way to combine multiple PDF reports into one combined PDF file with bookmarks and page numbering. At these times, we have certain restrictions where a SAS user depends upon a third-party tool. These third-party tools are often expensive or at times not validated enough to be used on a regular basis. As a SAS user one can accomplish certain tasks like combine a PDF or RTF and etc without depending on third party tool. SAS Private JRE is a combination of SAS and JAVA technology which will help the user and the organization to accomplish the task by not going for additional purchase of license or using any other tedious manual processes. The advantage of SAS Private JRE is the platform independence. The only necessity is that the user who creates the program or subroutine needs to have knowledge about JAVA, but the end user can use it freely without having a knowledge about JAVA. Using SAS Private JRE, this paper demonstrates combining multiple PDF documents into a single PDF file. This can be further extended to combining RTF files. There is also a lot of potential for SAS Private JRE to be used for other applications within the SAS programming world.

REFERENCES

SAS Private JRE:

http://support.sas.com/kb/38/518.html

https://support.sas.com/en/documentation/third-party-software-reference/9-4/support-for-java.html

JAVA JDK Download: https://www.oracle.com/technetwork/java/javase/downloads/index.html

SAS Private JRE- Installation: http://support.sas.com/kb/32/004.html

Itextpdf Package download link: http://www.java2s.com/Code/Jar/i/Downloaditextpdf540jar.htm

CONTACT INFORMATION

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

Praveen Kumar J
Ephicacy Lifescience Analytics
Bangalore, India
Praveen.kumar@ephicacy.in