Using SAS® to Control and Automate a Multi SAS Program Process

Patrick Halpin November 2008

What are we covering today

- A little background on me
- Some quick questions
- How to use “Done” files
    - Use a simple example
- Quick Review
- Questions

A little background about me

- Using SAS since 1996
- Certified base and advanced SAS programmer.
- Currently working at dunnhumbyUSA in Cincinnati Ohio
- Work has covered pricing and elasticity, statistical analysis, modeling, optimization, reporting and communications and media.
- Experience in banking, gaming and transportation industries.
- B.S in mathematics at Elizabethtown College
- M.S. in Mathematics/Operations Research at The College of William and Mary.

Questions for the audience

- How many times has one sat around waiting for a program to end to kick off the next one?
- What if you don't know when it will finish?
- What if you have multiple dependent programs?
- Then this presentation may help

Don’t worry this presentation has been through QA

To Control and Automate a Multi SAS Program Process one can use SAS

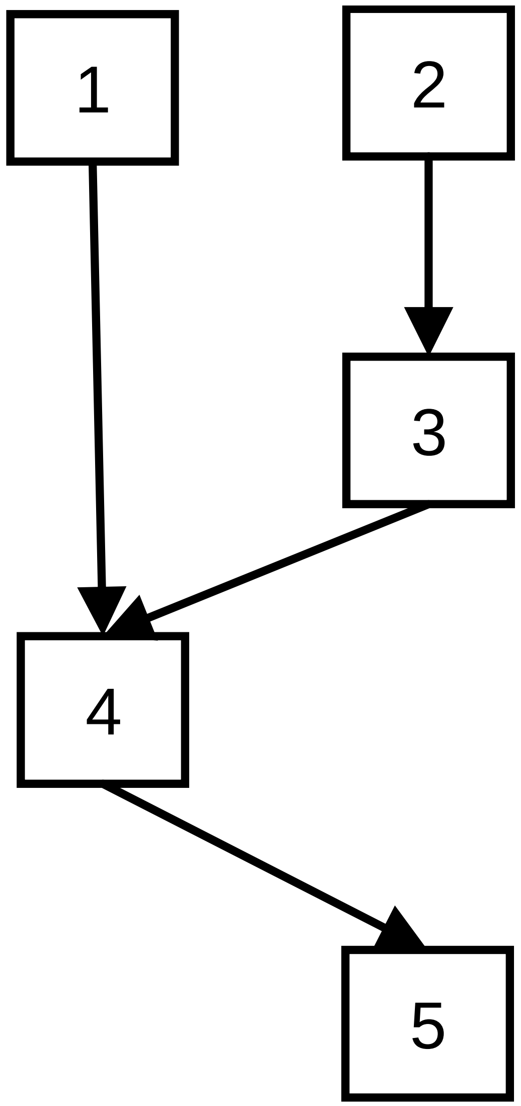
- Often times a project is comprised of many SAS® programs that need to run in a given order.
- Additionally, the SAS programs are dependent on previous SAS programs completing.
- One way to accomplish and automate this task is to use “Done” files and the SLEEP command in SAS.
- “Done” files are files that are created when a SAS program finishes.
- SAS programs dependent on previous jobs finishing look for those “Done” files to be created.
- Once the “Done” files are created the next SAS job in the process will start.

To control and automate the process there are 4 main steps

- Understanding the program flow
- Step 1 - Creating the UNIX shell script
- Step 2 - Creating “Done” files
- Step 3 - Using the SLEEP command
- Step 4 - Putting it all together
- We will use a simple 5 program example case to help explain these steps

It is vital to understand the program flow

- 1 and 2 can run together
- 3 needs 2 to finish
- 4 needs 1 and 3 to finish
- 5 needs 4 to finish



Step 1: A shell script will control out program flow

```
# UNIX Script
#! /bin/ksh
cd /Program_Directory
sas prog_1.sas &
sas prog_2.sas
sas prog_3.sas
sas prog_4.sas
sas prog_5.sas
```

1. Prog_1 and Prog_2 will run at the same time
2. Prog_3 runs when Prog_2 completes
3. Prog_4 runs when Prog_3 completes
4. Prog_5 runs when Prog_4 completes
5. Point 3 is **NOT** what we want to do
6. Prog_4.sas must start after **BOTH** prog_3.sas **AND** prog_1.sas
7. We need "Done" files and the SLEEP command

Step 2: “Done” Files are used to tell us about the status of a SAS program

- For our example we will create “Done” files when a SAS program has finished running
- We will need to add code to the start and end of the SAS programs
- At the start one deletes an existing “Done” files
- At the end one creates the “Done” file when the program is complete

Step 2: We need code to delete any existing “Done” files

```
/* Deletes an existing “Done” files */
Libname lib_done “/program_directory/”;
  %macro delete_done(var1);
  %if  %sysfunc(exist(lib_done.done_file&var1)) %then %do;
    proc datasets lib=lib_done nolist;
      delete done_file&var1;
    run;
  %end;
  %mend delete_done;
  %delete_done(program_name);
  run;
```

Step 2: We need code to create “Done” files

```
/* Creates a “Done” files */
%macro create_done(var1);
   Data lib_done.done_file&var1;
      done=”YES”;
      output;
   run;
   %mend create_done;
   %create_done(program_name);
   run;
```

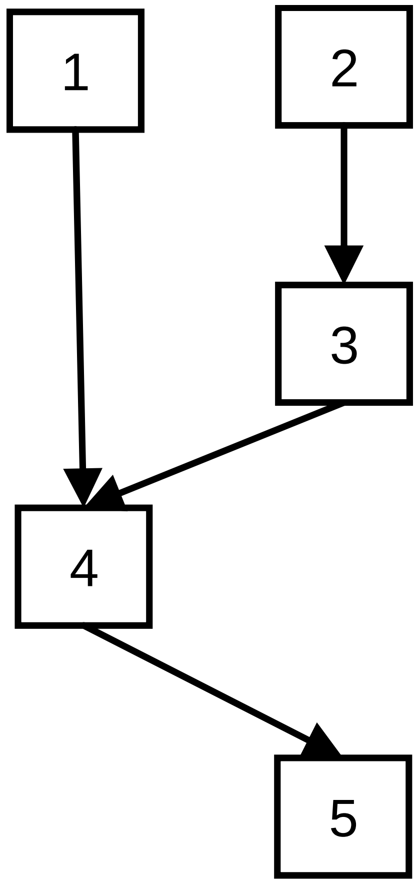
Step 3: Using the sleep command to pause a SAS program

- We will use the SLEEP command to pause a SAS program.
- This program will be paused until precedent program/process is complete.
- The syntax for the Sleep command is simple
    - SLEEP(*n*<,*unit*>)
- This command is often used with DDE an open up excel files
- Here is an example where the SAS program will pause for 10 seconds

```
 Data _null_;
        X=Sleep(10,1);
run;
```

Step 4: We need to put all the step together to make this work

- **Create the UNIX script**
  - #! /bin/ksh
  - cd /Program_Directory
  - sas prog_1.sas &
  - sas prog_2.sas
  - sas prog_3.sas
  - sas prog_4.sas
  - sas prog_5.sas
- Add create/delete “Done” file code to programs 1, 3, 4 and 5


$$\begin{tikzcd}
\boxed{1} \arrow[dd] & \boxed{2} \arrow[d] \\
 & \boxed{3} \arrow[dl] \\
\boxed{4} \arrow[dr] & \\
 & \boxed{5}
\end{tikzcd}$$

Step 4: More code is needed in program 4

- Program 4 will look for the “Done” files from 1 and 3.
- Need to add a SLEEP command and loop to check for those “Done” files
- Make loop time long enough that 1 and 3 should have completed
  - We will check every 2 minutes
- Have program time out with an error message if the “Done” file is not created in a reasonable amount of time
  - If after the 2 hours “Done” files are not found 4 will not run
- Code will be added to 5 to stop it if 4 does not run
- Run the UNIX script

Step 4: Code added at the top of program 4 to check for both done files

```
%global counter max;
%let counter=0;   %let max=120; /* minutes to check */
%macro done_check;
%if ((%sysfunc(exist(lib_done.done_prog_1)))=0 or
    (%sysfunc(exist(lib_done.done_prog_3)))=0) %then %do;
    %do %until (((%sysfunc(exist(lib_done.done_prog_1)))=1 and
           (%sysfunc(exist(lib_done.done_prog_3)))=1)
          or (&counter+0 > &max+0));
       data _null_;
           sleep_time=sleep(60,1); *** sleep for a minute ***;
        run;
        %let counter=%eval(&counter.+1); *** increment counter ***;
    %end;
%end;
```

Step 4: Code to stop program 4 if "done" files not found in time

```
*** Stop program if time out ***;
  %if &counter+0 > &max+0 %then endsas;
  %mend done_check;
  %done_check; run;
```

Step 4: Check at the start of program 5 to confirm 4 finished successfully

```
/* Code added to the top of prog_5 */
/* To stop it if 4 does not run */

   Libname lib_done “/program_directory/”;
   %macro check(var1);
   %if sysfunc(exist(lib_done.done_file&var1))=0 %then
    endsas;
   %end;
   %mend check;
   %check(prog_4);
   run;
```

A quick review of the process

- Know the program flow
- 4 main steps
  - Shell script
  - “Done” files
  - Sleep command
  - Put it all together
    - Adding code at the start and end of programs
    - To check to see when and how programs ended

“Done” files and the Sleep command can Control and Automate a Multi SAS Program Process

- Projects often consist of multiple SAS programs, the dependences and order of processing can become increasingly complex.
- What we have discussed above is a simple solution that can be used anywhere SAS exists.
- Using just a UNIX shell script and a couple of commands within your programs, processes can be streamlined.
- Many more bells and whistles can be added to the UNIX script and the “Done” files to build out the process further to increase robustness and efficiency.

Questions

Thank You