JCL: Kicking Things Off

Some advanced features that might come in handy

9 steps 60 minutes

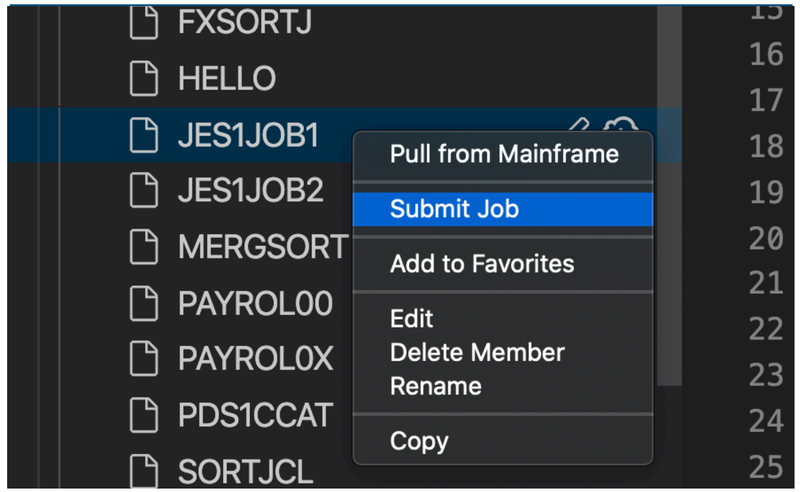
THE CHALLENGE

We use JCL (Job Control Language) in z/OS to let the system know what tasks it should perform. The JCL describes the programs and data used, as well as what to do with it when it's done processing.

A task or series of tasks written in JCL is submitted to the system as a "Job", and as the job gets handled, we can view its output. All of this can be done through VS Code.

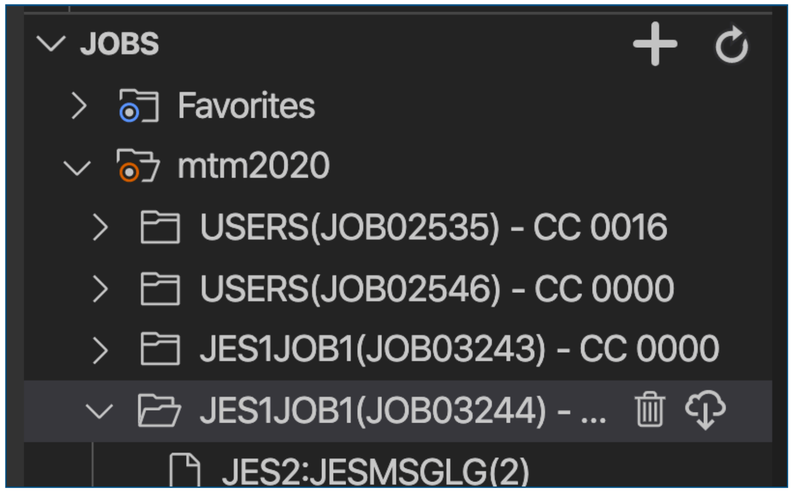
BEFORE YOU BEGIN

You should have VS Code fully installed and configured before you begin. We suggest you attempt and finish both "Files for Miles" and "Thanks for the Memberies" before starting these JCL challenges, but if you're feeling adventerous, then go for it!



1. RUN THE JES1JOB1 JCL

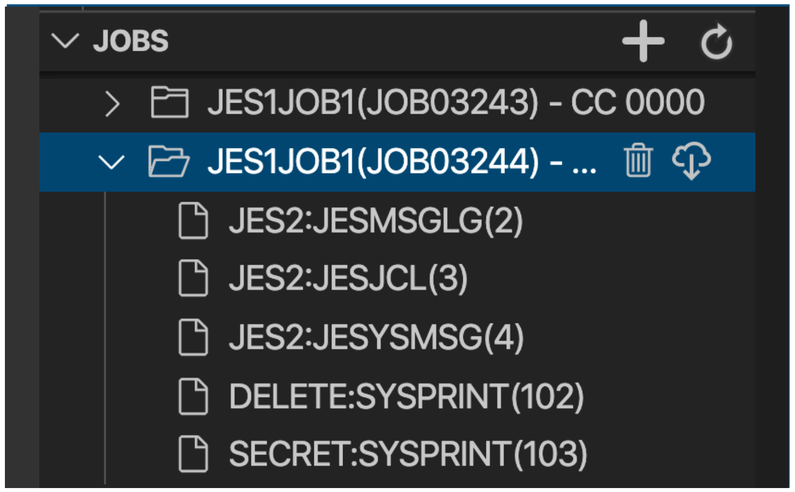
Find the JES1JOB1 file in **MTM2020.PUBLIC.JCL** and copy it into your own JCL data set (ZXXXXX.JCL), giving it the same name. Right-click on the copy in your JCL data set and select "Submit Job". The job should have run with no problems.



2. CHECK OUT JOBS

Verify that the job ran correctly by looking in the "Jobs" bar, which should be at the bottom on the left side unless you moved it. There should be a profile already defined in there, which you did in Step 12 of the first VS Code Challenge, "Let's Get Connected".

Protip: Did you see that box pop up in the bottom right corner of VS Code? You can also just click on that, and it'll take you directly to the job output. Kinda nice!

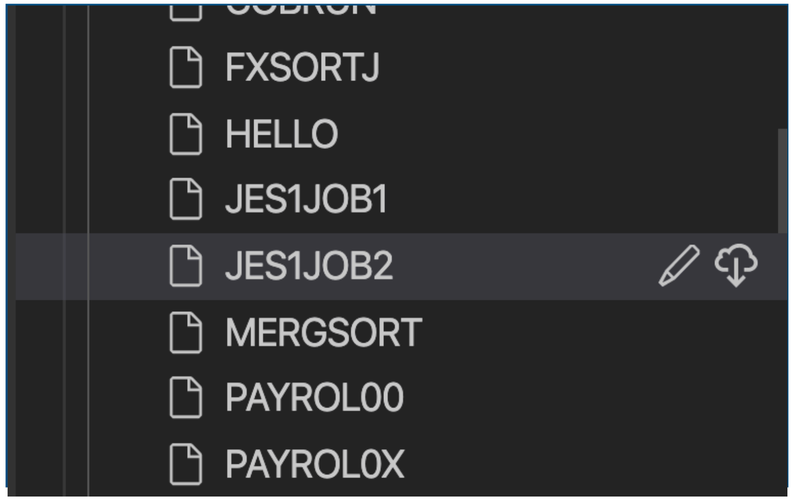


3. FIND THE JES1JOB1 JOB

Open up the profile and look for JES1JOB1. Remember that you may need to hit the Refresh icon to see the latest updates in here, so give it a few seconds and look for something like the screenshot above.

If you still don't see any output, right-click on the profile name, and set Job Prefix to blank, and Job Owner to your ID (ZXXXXX). You may have a filter set that's not showing any of your output.

```
1 //JES1JOB1 JOB 1
2 //DELETE   EXEC PGM=IDCAMS
3 //SYSPRINT DD SYSOUT=*
4 //SYSIN    DD *,SYMBOLS=CNVTSYS
5 //SECRET EXEC PGM=ICEGENER
6 //SYSPRINT DD SYSOUT=*
7 //SYSUT1   DD DSN=MTM2020.PUBLIC.INPUT(SECRET),DISP=SHR
8 //SYSUT2   DD DSN=&SYSUID..SECRET.WORD,DISP=(NEW,CATLG,DELETE),
  //         UNIT=SYSALLDA,SPACE=(TRK,1),
  //         DCB=(LRECL=80,DSORG=PS,RECFM=F)
  IEFC653I SUBSTITUTION JCL - DSN=[illegible],DISP=(NEW,CAT
  DCB=(LRECL=80,DSORG=PS,RECFM=F)
9 //SYSIN    DD DUMMY
```




```
//JES1JOB2 JOB 1
//*
//*  JCL line 5 needs a SECRET= value to execute
//*
//SECRET   EXEC PGM=IEBGENER
//SYSPRINT DD DUMMY
//SYSUT1   DD DSN=MTM2020.PUBLIC.WORK(&SECRET),DISP=SHR
//SYSUT2   DD DSN=&SYSUID..OUTPUT(JES1JOB2),DISP=SHR
//SYSIN    DD DUMMY
```

4. FIND THE SECRET WORD

Click on the arrow to open up the output for JES1JOB1. You will probably find five different pieces of output in there.

Somewhere in there (as well as in the JCL you submitted) there is an output data set where a secret word got stored.

Open up ***that*** data set and look for the secret word inside. It's a bit of a scavenger hunt, but when you find it, it'll be obvious.

5. COPY OVER JES1JOB2

Copy JES1JOB2 over from the MTM2020.PUBLIC.JCL data set, putting a copy in your own JCL data set with the same name. Open it up and look at the JCL in here.

6. READ THE COMMENTS

Look at lines 2-4. They start with //* and may show up as green, meaning they are comments. Those comments are there for you, not the system, so read them and figure out what you have to do.

It's easy to think that comments are not as important as code, but often, they reveal the true intention of the original author, which can be very helpful when trying to make changes, or fix problems. They can be used to spell out how the program is meant to be used, what its limitations are, or who to contact in case there is a problem.

For this particular problem, they're letting you know that in order for this program, you need to find a secret word and put it on Line 5 of the JCL.

"WHAT ARE THE DIFFERENT OUTPUTS FOR?"

DD statements are the stars of JCL statements. Understand them, and your life will become easier. Master them and you will become a JCL Genius.

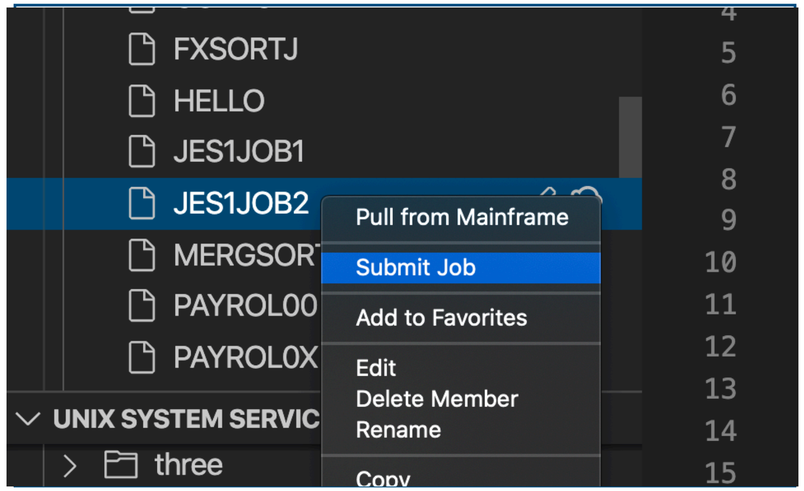
Part by part, DD sets it up by saying "This is a Data Definition", and then spells out where that data goes in the DSN section. There are other places data can go, including temporary files and output spools, but in our example, these are data set members (you can tell because they're in (parenthesis). Lastly is the DISP statement, which determines whether it expects that input or output to exist when it starts, how to handle it, and what to do with it once it's done using it.

```
//JES1JOB2 JOB 1
//*
//*  JCL line 5 needs a SECRET= value to execute
//*
//SETVAR   SET SECRET=MAINFRAME
//SECRET   EXEC PGM=IEBGENER
//SYSPRINT DD DUMMY
//SYSUT1   DD DSN=MTM2020.PUBLIC.WORK(&SECRET),DISP=SHR
//SYSUT2   DD DSN=&SYSUID..OUTPUT(JES1JOB2),DISP=SHR
//SYSIN    DD DUMMY
```

7. MAKE THE UPDATE

Update the line with the secret word. Don't put in any quotes, dashes, or anything fancy. Just fill in the empty spaces after the equals sign with the secret word you found in Step #4.

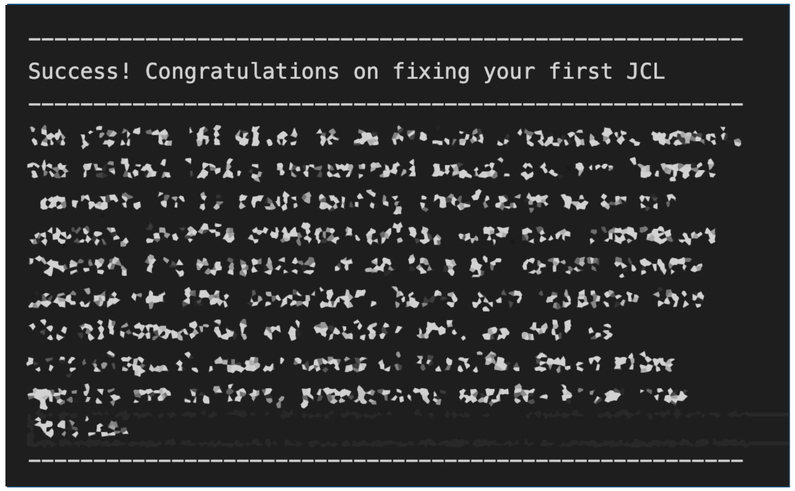
(By the way, the secret word is *NOT* mainframe, that's just an example)



8. SAVE AND SUBMIT

Save the file so your update gets uploaded, then right-click on the file and select "Submit Job". Same as before, check in Jobs for your output, and look for JES1JOB2.

If there are errors, then perhaps you didn't enter the right secret word, didn't save the file after making updates, or didn't run the correct version. When there are multiple versions, the lower ones are usually the most recent.



9. CHECK THE OUTPUT

See if you can figure out where that job's output went. If all went well, you'll see a nice little message based around that secret word. Congratulations!

To validate and mark as complete, open up MTM2020.PUBLIC.JCL and look for CHK. Right-click on it, and then select "Submit Job" to validate it.

NICE JOB! LET'S RECAP

You've taken a look at some JCL, and run it. You looked at the output, and used what you saw in there to make edits to another piece of JCL, which you successfully submitted.

So now you know how to run JCL and look at the output. In the next JCL challenge, we'll take a deeper look at what some of those crazy looking lines in there actually mean.

NEXT UP...

Got a grasp on submitting jobs and looking at the output? Great! We've got a great follow-up challenge called JCL2 that we think would hit the spot right about now.