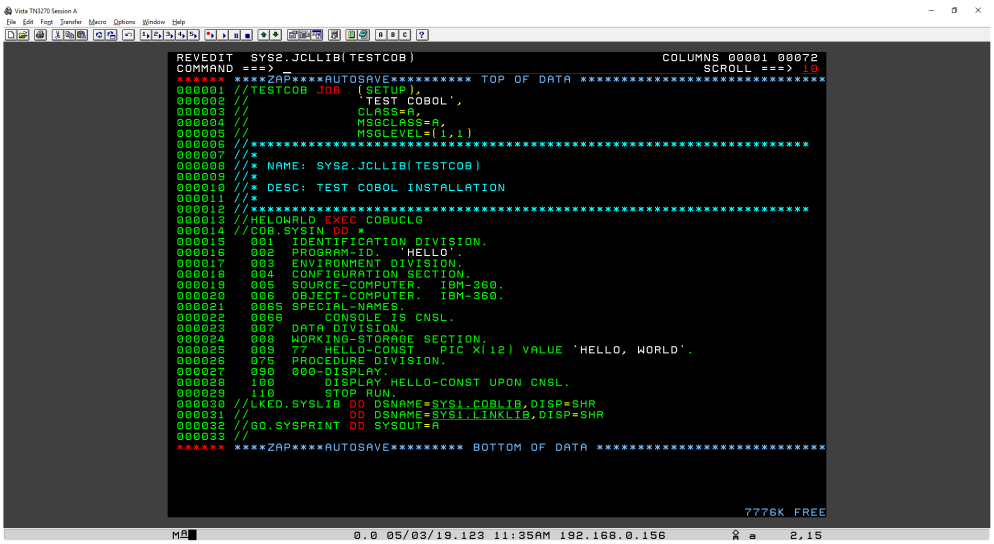
MOM Pt. 003: Getting to know COBOL – Compiling

kevindurant Mainframe, My Own Mainframe May 3, 2019 2 Minutes

Today I'm going to get to know COBOL. I've already used COBOL in IBM's Master the Mainframe 2018 contest, but this time I'm working in MVS 3.8!

What am I planning to do here? Well, before I start learning COBOL I want to know how I can edit, compile and run it on my own mainframe. MVS3.8 provides a small piece of test code in **SYS2.JCLLIB** called **TESTCOB**.

So I go to **RFE.** Then I need to navigate in the **DSLIST** utility, so that means I enter **3.4** to quickly get there. The data set name prefix is **SYS2.JCLLIB**. When **e**diting, I am able to scroll down and **e**dit **TESTCOB**.



So this is **TESTCOB**. Normally any piece of **COBOL production code** has **four divisions**.

- IDENTIFICATION DIVISION
  - Here we identify the COBOL program, the name of the program itself. Typically, the program documentation goes here.
- ENVIRONMENT DIVISION
  - Here we link the program with external data sets.
- DATA DIVISION

- Here we define the local variables, exclusive to this program only.

- PROCEDURE DIVISON
  - Here we code all our logic.

As you can see on the image below, **TESTCOB** indeed has the four divisions.

```
REVEDIT  SYS2.JCLLIB(TESTCOB)                                  COLUMNS 00001 00072
COMMAND ===> _                                                 SCROLL ===> 10
****** ****ZAP****AUTOSAVE********** TOP OF DATA ***************************
000001 //TESTCOB JOB  (SETUP),
000002 //             'TEST COBOL',
000003 //             CLASS=A,
000004 //             MSGCLASS=A,
000005 //             MSGLEVEL=(1,1)
000006 //*********************************************************************
000007 //*
000008 //* NAME: SYS2.JCLLIB(TESTCOB)
000009 //*
000010 //* DESC: TEST COBOL INSTALLATION
000011 //*
000012 //*********************************************************************
000013 //HELOWRLD EXEC COBUCLG
000014 //COB.SYSIN DD *
000015    001  IDENTIFICATION DIVISION.
000016    002  PROGRAM-ID.   'HELLO'.
000017    003  ENVIRONMENT DIVISION.
000018    004  CONFIGURATION SECTION.
000019    005  SOURCE-COMPUTER.   IBM-360.
000020    006  OBJECT-COMPUTER.   IBM-360.
000021    0065 SPECIAL-NAMES.
000022    0066      CONSOLE IS CNSL.
000023    007  DATA DIVISION.
000024    008  WORKING-STORAGE SECTION.
000025    009  77   HELLO-CONST    PIC X(12) VALUE 'HELLO, WORLD'.
000026    075  PROCEDURE DIVISION.
000027    090  000-DISPLAY.
000028    100       DISPLAY HELLO-CONST UPON CNSL.
000029    110       STOP RUN.
000030 //LKED.SYSLIB DD DSNAME=SYS1.COBLIB,DISP=SHR
000031 //             DD DSNAME=SYS1.LINKLIB,DISP=SHR
000032 //GO.SYSPRINT DD SYSOUT=A
000033 //
****** ****ZAP****AUTOSAVE******** BOTTOM OF DATA ************************
```

Now what have I highlighted in pink? Those are sections. Sections can be omitted but they have chosen not to in this case. At line **000027** you can see **000-DISPLAY**, that is a routine. The next two lines are **statements/sentences** inside a **paragraph**. These state that the machine needs to **DISPLAY** the **HELLO-CONST** in **CNSL**. The **HELLO-CONST** is defined at line **000025** in the data division. **CNSL** is an alias for **CONSOLE** defined in the **CONFIGURATION** section. The code pretty much speaks for itself right? Display **HELLO, WORLD** in the console and then **STOP** running.

Now what is **HELLO-CONST PIC X(12) VALUE 'HELLO, WORLD'**? This means that **HELLO-CONST** can carry 12 bytes. 1 byte equals to 1 character in an **EBCDIC** mainframe. You can see **X** as a template, in theory **HELLO-CONST** is **XXXXXXXXXXXX**. We fill those X's with 'HELLO, WORLD', X means the repetition factor meaning X is repeated 12 times.

```
SPOILER ALERT: change MSGCLASS=A to MSGCLASS=H on line 000004!
```

When you're finished checking this file, enter the primary command **save** to save your changes. To execute your program, enter the primary command **submit**.

```
JOB TESTCOB(JOB00002) SUBMITTED
```

To check our output we need to go back to the root **RFE** and navigate to the **OUTLIST (3.8)**. Select the correct job by using the line command **s**.

Kevin's Coding Blog is experiencing technical difficulties

It’s not here? Where is it? The COBOL job isn’t outputting to the OUTLIST?

Please allow me to add some Google tags for those with the same issue: cobol mainframe MVS 3.8 tk4 no job output help sysprint please google.

I forgot to change the **MSGCLASS** to **H**. This means you want the output from the compiler to be **held**. Now let’s submit it again and check the output.

```
JOB TESTCOB(JOB00025) SUBMITTED
```

Yes I know, 25… I’ve been struggling. Again the job output was empty, but after entering the primary command **ST *** I can see all the jobs.

TESTCOB is there! Now I’ll select with **s** and see what it did.

There it is! **HELLO, WORLD**. We have done it!

Next up: our own custom mix of **JCL** and **COBOL**!

Are you experiencing any technical difficulties?

Tell me all about it!

Tagged: 3.8j, cobol, Hercules, JCL, mainframe, MVS, MVS3.8, programming, Turnkey

Published by kevindurant

View all posts by kevindurant

‹ MOM Pt. 002: Custom NETSOL Sign-On Screen

MOM Pt. 004: Writing simple COBOL on MVS3.8 tk4- ›

5 thoughts on “MOM Pt. 003: Getting to know COBOL – Compiling”

Carsten Spräner

February 12, 2023 at 9:30 pm

Thank you for this blog! They opened a whole new world to me. A world I only knew from my colleagues speaking about job queues, JCLs, and data sets. As a Java programmer it is really good to know what they are talking about.

↩ Reply

John Doe

March 9, 2021 at 2:36 am

The test script throws several errors. We Get:
03.11.20 JOB 31 IEF677I WARNING MESSAGE(S) FOR JOB TESTCOB ISSUED

03.11.20 JOB 31 $HASP373 TESTCOB STARTED – INIT 1 – CLASS A – SYS TK4-
03.11.20 JOB 31 IEF403I TESTCOB – STARTED – TIME=03.11.20
03.11.20 JOB 31 IEC130I SYSPUNCH DD STATEMENT MISSING
03.11.20 JOB 31 IEC130I SYSLIB DD STATEMENT MISSING
03.11.20 JOB 31 IEC130I SYSPUNCH DD STATEMENT MISSING
03.11.20 JOB 31 IEFACTRT – Stepname Procstep Program Retcode
03.11.20 JOB 31 TESTCOB HELOWRLD COB IKFCBL00 RC= 0000
03.11.20 JOB 31 TESTCOB HELOWRLD LKED IEWL RC= 0000
03.11.20 JOB 31 +HELLO, WORLD
03.11.20 JOB 31 TESTCOB HELOWRLD GO PGM=*.DD RC= 0000
03.11.20 JOB 31 IEF404I TESTCOB – ENDED – TIME=03.11.20
03.11.20 JOB 31 $HASP395 TESTCOB ENDED

↪ Reply

Georges
October 29, 2019 at 3:29 am

Hi Kevin,
In ch4 part3 I had the same issue, but Bob L from the mainframe slack gave me the answer: In SDSF try: prefix * and then: owner Z##### the name of the owner of the work.
This should work in your MOM too!
You tell me!
Georges

↪ Reply

kevindurant
October 29, 2019 at 7:33 am

Hi Georges,

Interesting! Thanks for sharing! I'll try this out, I had some trouble in the last MTM 19 challenges I have blogged about. I didn't see my outputs but I knew what I was doing so I didn't really need it. I guess you could say I was being lazy, heheh.

Cheers!
– Kevin

↪ Reply

Marco
May 6, 2019 at 2:23 pm

Things would be simpler if you change the jcl name from TESTCOB to something like HERC01C, where HERC01 is the user profile you are using

↪ Reply

Leave a Reply

Your email address will not be published. Name, email and website not required.

Comment

Name

Email

Website

Save my name, email, and website in this browser for the next time I comment.

Post Comment

Recent Posts

IBM MTM 2020: Blog update

Tackle Master the Mainframe using Zowe CLI and VSCode

IBM MTM 2019: Final Words

IBM MTM 2019: Part Three – Challenge #15

IBM MTM 2019: Part Three – Challenge #14

Receive e-mail updates!

Email*

Submit

Search

Search …

Recent Comments

Britt Briseno on **IBM MTM 2019: Part Two – Challenge #13**

Howard Lewin on **IBM MTM 2019: Part One – Challenge #01**

Charlie Isabel on **IBM MTM 2018: Part Two – Challenge #05**

Zane Cadman on **IBM MTM 2018: Part Three – Challenge #14**

Jake Wootton on **IBM MTM 2019: Final Words**

Archives

September 2020

April 2020

January 2020

December 2019

November 2019

October 2019

September 2019

July 2019

May 2019

March 2019

January 2019

December 2018

November 2018

October 2018

September 2018

Categories

IBM Master the Mainframe 2018

IBM Master the Mainframe 2019

IBM Master The Mainframe 2020

Mainframe

My Own Mainframe

Zowe

Meta

Log in

Entries RSS

Comments RSS

WordPress.org