

> ### *Try it Option Online*
>
> *We have set up the COBOL Programming environment online, so that you can compile and execute all the available examples online. It gives you confidence in what you are reading and enables you to verify the programs with different options. Feel free to modify any example and execute it online.*
>
> *Try the following example using our online compiler available at CodingGround*

```
IDENTIFICATION DIVISION.
PROGRAM-ID. HELLO.

PROCEDURE DIVISION.
   DISPLAY 'Hello World'.
STOP RUN.
```

> *For most of the examples given in this tutorial, you will find a Try it option in our website code sections at the top right corner that will take you to the online compiler. So just make use of it and enjoy your learning.*

Installing COBOL on Windows/Linux

There are many Free Mainframe Emulators available for Windows which can be used to write and learn simple COBOL programs.

One such emulator is Hercules, which can be easily installed on Windows by following a few simple steps as given below:

- Download and install the Hercules emulator, which is available from the Hercules' home site: www.hercules-390.eu
- Once you have installed the package on Windows machine, it will create a folder like **C:/hercules/mvs/cobol**.
- Run the Command Prompt *CMD* and reach the directory C:/hercules/mvs/cobol on CMD.
- The complete guide on various commands to write and execute a JCL and COBOL programs can be found at:

  www.jaymoseley.com/hercules/installmvs/instmvs2.htm

Hercules is an open-source software implementation of the mainframe System/370 and ESA/390 architectures, in addition to the latest 64-bit z/Architecture. Hercules runs under Linux, Windows, Solaris, FreeBSD, and Mac OS X.

A user can connect to a mainframe server in a number of ways such a thin client, dummy terminal, Virtual Client System *VCS*, or Virtual Desktop System *VDS*. Every valid user is given a login id to enter into the Z/OS interface *TSO/EorISPF*.

Compiling COBOL Programs

In order to execute a COBOL program in batch mode using JCL, the program needs to be compiled, and a load module is created with all the sub-programs. The JCL uses the load module and not the actual program at the time of execution. The load libraries are concatenated and given to the JCL at the time of execution using **JCLLIB** or **STEPLIB**.

There are many mainframe compiler utilities available to compile a COBOL program. Some corporate companies use Change Management tools like **Endevor**, which compiles and stores

every version of the program. This is useful in tracking the changes made to the program.

```
//COMPILE   JOB ,CLASS=6,MSGCLASS=X,NOTIFY=&SYSUID
//*
//STEP1     EXEC IGYCRCTL,PARM=RMODE,DYNAM,SSRANGE
//SYSIN     DD DSN=MYDATA.URMI.SOURCES(MYCOBB),DISP=SHR
//SYSLIB    DD DSN=MYDATA.URMI.COPYBOOK(MYCOPY),DISP=SHR
//SYSLMOD   DD DSN=MYDATA.URMI.LOAD(MYCOBB),DISP=SHR
//SYSPRINT  DD SYSOUT=*
//*
```

IGYCRCTL is an IBM COBOL compiler utility. The compiler options are passed using the PARM parameter. In the above example, RMODE instructs the compiler to use relative addressing mode in the program. The COBOL program is passed using the SYSIN parameter. Copybook is the library used by the program in SYSLIB.

Executing COBOL Programs

Give below is a JCL example where the program MYPROG is executed using the input file MYDATA.URMI.INPUT and produces two output files written to the spool.

```
//COBBSTEP  JOB CLASS=6,NOTIFY=&SYSUID
//
//STEP10    EXEC PGM=MYPROG,PARM=ACCT5000
//STEPLIB   DD DSN=MYDATA.URMI.LOADLIB,DISP=SHR
//INPUT1    DD DSN=MYDATA.URMI.INPUT,DISP=SHR
//OUT1      DD SYSOUT=*
//OUT2      DD SYSOUT=*
//SYSIN     DD *
//CUST1     1000
//CUST2     1001
/*
```

The load module of MYPROG is located in MYDATA.URMI.LOADLIB. This is important to note that the above JCL can be used for a non-DB2 COBOL module only.

Executing COBOL-DB2 programs

For running a COBOL-DB2 program, a specialized IBM utility is used in the JCL and the program; DB2 region and required parameters are passed as input to the utility.

The steps followed in running a COBOL-DB2 program are as follows:

- When a COBOL-DB2 program is compiled, a DBRM *DatabaseRequestModule* is created along with the load module. The DBRM contains the SQL statements of the COBOL programs with its syntax checked to be correct.
- The DBRM is bound to the DB2 region *environment* in which the COBOL will run. This can be done using the IKJEFT01 utility in a JCL.
- After the bind step, the COBOL-DB2 program is run using IKJEFT01 *again* with the load library and the DBRM library as the input to the JCL.

```
//STEP001  EXEC PGM=IKJEFT01
//*
//STEPLIB  DD DSN=MYDATA.URMI.DBRMLIB,DISP=SHR
//*
//input files
//output files
//SYSPRINT DD SYSOUT=*
//SYSABOUT DD SYSOUT=*
//SYSDBOUT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DISPLAY  DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
```

```
//SYSTSIN  DD *
    DSN SYSTEM(SSID)
    RUN PROGRAM(MYCOBB) PLAN(PLANNAME) PARM(parameters to cobol program) -
    LIB('MYDATA.URMI.LOADLIB')
    END
/*
```

In the above example, MYCOBB is the COBOL-DB2 program run using IKJEFT01. Please note that the program name, DB2 Sub-System Id *SSID*, and DB2 Plan name are passed within the SYSTSIN DD statement. The DBRM library is specified in the STEPLIB.

Try it Option Online

You really do not need to set up your own environment to start learning COBOL programming language. Reason is very simple, we have already set up COBOL Programming environment online, so that you can compile and execute all the available examples online at the same time when you are doing your theory work. This gives you confidence in what you are reading and to check the result with different options. Feel free to modify any example and execute it online.

Try the following example using our **Try it** option available alongside the code in our website.

```
IDENTIFICATION DIVISION.
PROGRAM-ID. HELLO.
PROCEDURE DIVISION.
DISPLAY 'Hello World'.
STOP RUN.
```

When you compile and execute the above program, it produces the following result:

```
Hello World
```

For some of the examples given in this tutorial, you will find a **Try it** option in our website code selections at the to right corner that will take you to the online compiler. So just make use of it and enjoy your learning. Try it option would work only with the code compatible with OpenCOBOL. The programs that require JCL *Inputfile OutputfileorParameters* for execution would not run on Tryit option.