z/OS Introduction and Workshop

Job Control Language (JCL)

Unit objectives

After completing this unit, you should be able to:

- Explain the purpose of JCL
- Recognize JCL Statements & Fields
- List the most significant JCL reserved words
- Explain the JCL relationship to program file names
- Explain DD Operation and I/O Device Independence
- Find sources of information to advance JCL skill level

“In the beginning..”

Mainframes prior to S/360 were designed for scientific application number crunching.

The original S/360 hardware was designed from the ground up to meet the needs of business where data throughput capability was greater than the speed of number crunching.

The original OS/360 needed to work with **many newly planned Input and Output, I/O, devices**, aka “peripherals”, to handle data throughput.

Business applications needed to be independent of any peripheral I/O device.

The S/360 and OS/360 design **required device-independent I/O methods**.

Fred Brooks managed development of System 360 which evolved into today's mainframe

Fred Brooks jokes about JCL saying,

- *“I always tell my students OS/360 Job Control Language is the **worst programming language** ever designed anywhere by anybody for any purpose and it was done under my management.”*

OS/360 JCL, “the Worst Language”

Done under my management “Fred Brooks”

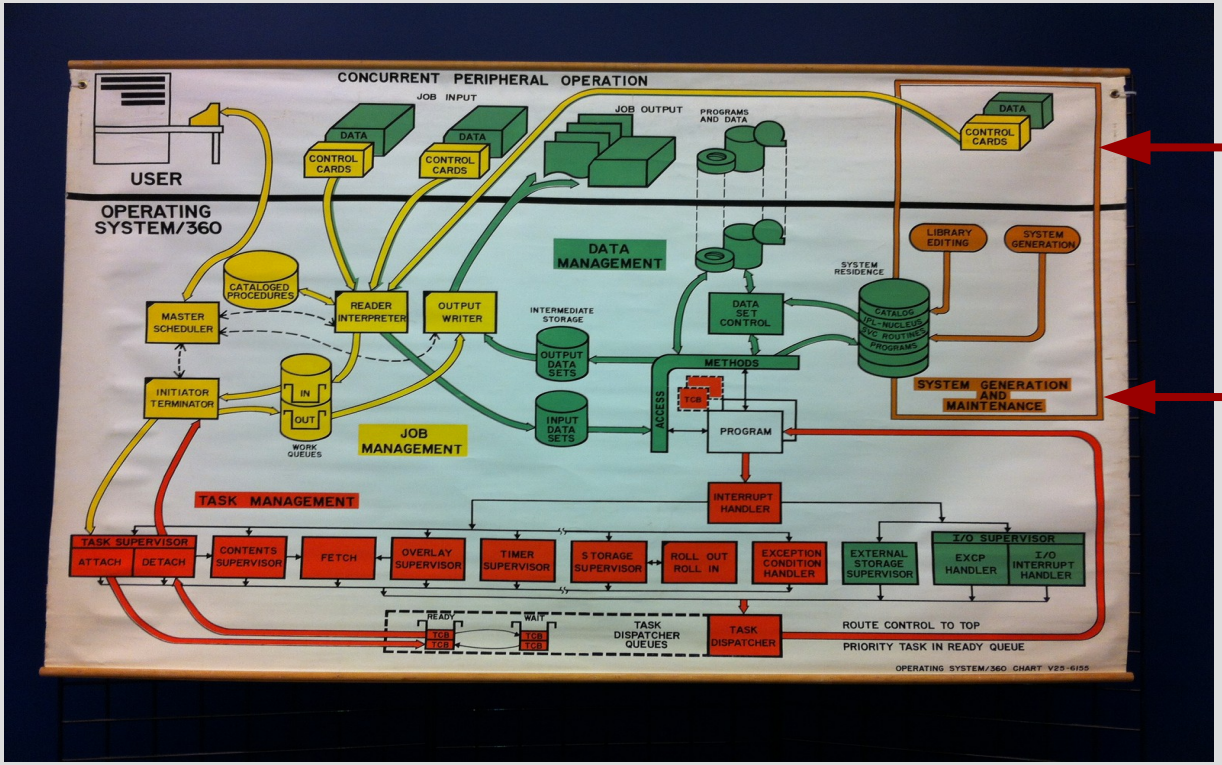
- **One job language for all programming languages**
- Like Assembler language, rather than PL/I, etc.
- But not exactly like: card-column dependent
- Too few verbs
- Declarations do verbish things, via parameters
- Awkward branching
- No clean iteration
- No clean subroutine call

Basic problem was pedestrian vision

- We did not see it as a schedule-time programming language, but as a “few control cards”
  It was not *designed*, it just grew as needs appeared.

“The Purpose of JCL”

JCL provided for the **requirement** of business applications to be **independent of the I/O devices**



I/O Devices

Operating System

What made JCL the “worst” **language**?

The **L** in **JCL**

The idea of “**one job language for all programming languages**” was a genius idea

JCL is best thought of as a single mechanism to execute all programming languages

Sequential Stream of Statements

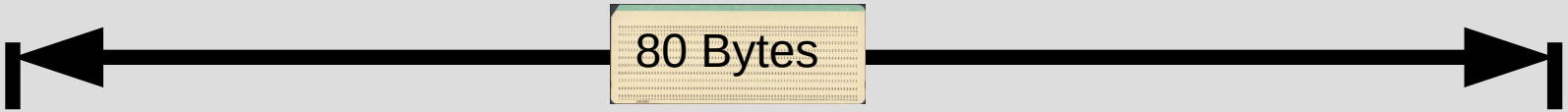
Job Control Language (JCL) is a sequential collection of **80** character records beginning with **//** which the operating system reads and interprets

JCL is used to

- Assign name and authority level
- Assign resources (programs, data, etc.) and services needed from the operating system to process a task

JCL can be viewed as a list of statements to be ‘submitted’ for background (batch) processing or ‘started’ for foreground (started task) processing

JCL Statement Fields



//NAME **OPERATION** **PARAMETER** **SEQ**

Parameter – Positional and Keyword details for the operation **

Ignored (73-80) Sequence numbers

** original JCL manual referred to PARAMETER field as OPERAND field

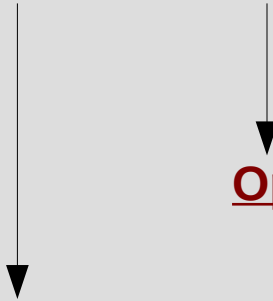
Operation – Type of statement
Most common are **EXEC** and **DD**

Name – Identifies the statement so that other statements and system can refer to it.
8 bytes or less

Identifier starts in column 1
- // (followed by name and/or operation)
- /* (delimiter – end of data)
- //* (comment)
- // (followed by all blanks – null..end of job)

JCL Execute Program Statement

//MYSTEP EXEC PGM=



Parameter named program for the execute operation

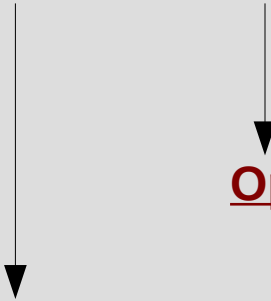
Operation is to execute

Name is a user selected “STEPNAME”

STEPNAME label identifies a specific EXEC statement

JCL Execute Program Statement

//MYSTEP EXEC PROC=



Parameter to exec a named JCL Procedure

Operation is to execute

Name is a user selected “STEPNAME”

PROC STEPNAME label identifies a specific EXEC statement

Most Significant JCL Reserved Words

```
//JOBNAME       JOB
//STEPNAME      EXEC
//DDNAME        DD
//*            … this is a comment statement
/*             … this indicates end of data
//             … this indicates end for JCL
```

JCL Data Definition (DD)

//DDNAME DD DISP=SHR,DSNAME=

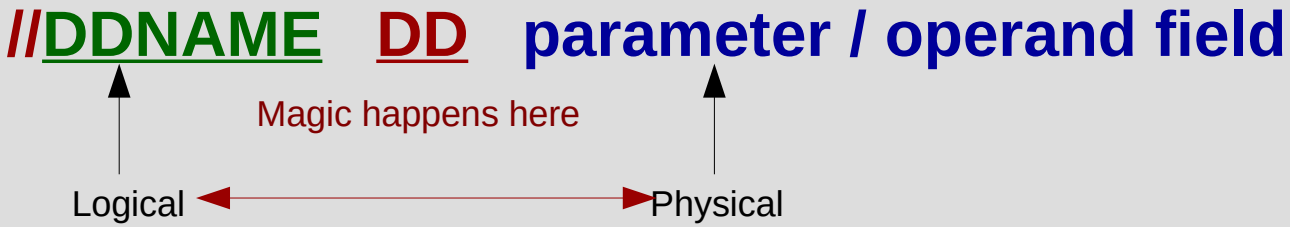
Parameters describe the input or output resource

Operation is **D**ata **D**efinition

Name must match spelling of a program file name

Each ddname must be unique within EXEC stepname

JCL Data Definition (DD)



Redirection magic from 1964 designed into OS/360

Allocation of system managed resources

JCL Statement Stream

```
//STEP1         EXEC   PGM=MYPGM1
//PGMI          DD     DSN=MY.INPUT.DATA,DISP=SHR
//PGMO          DD     DSN=MY.OUTPUT.DATA,DISP=SHR
//*****
//*  End STEP1 execution and Begin STEP2 execution
//*****
//STEP2         EXEC   PGM=SYSPGM1
//SYSI          DD     DSN=SYS.INPUT.DATA,DISP=SHR
//SYSO          DD     DSN=SYS.OUTPUT.DATA,DISP=SHR
```

A **JOB** is a **collection of related job STEPS** - identified by a **JOB** statement.

When JCL is submitted using **submit** command, the JCL needs a ***JOB*** statement

JOB statement can be **coded** or system will prompt to **generate** a **JOB** statement

JCL DD **Concatentation**

DD statement with a blank DDNAME is owned by previous DDNAME
MYPGM1 reads all 3 data sets associated with DDNAME PGMI
”**Concatentation**” of DDNAMEs

```
//STEP1     EXEC  PGM=MYPGM1
//PGMI      DD    DSN=MY.INPUT.DATA,DISP=SHR
//          DD    DSN=YOUR.DATA,DISP=SHR
//          DD    DSN=SYS.DATA,DISP=SHR
//PGMO      DD    DSN=MY.OUTPUT.DATA,DISP=SHR
```

JCL **Continuation**

Continuation of JCL operation statement is a comma followed by a space, then the next line begins with ***//*** - one space followed by additional parameters for the JCL operation

```
//STEP1     EXEC  PGM=MYPGM1
//PGMI      DD    DSN=MY.INPUT.DATA,
//   DISP=SHR
//PGMO      DD    DSN=MY.OUTPUT.DATA,
//   DISP=SHR
```

JCL, **J**ob **C**ontrol **L**anguage

Computer code that tells the operating system what to do.

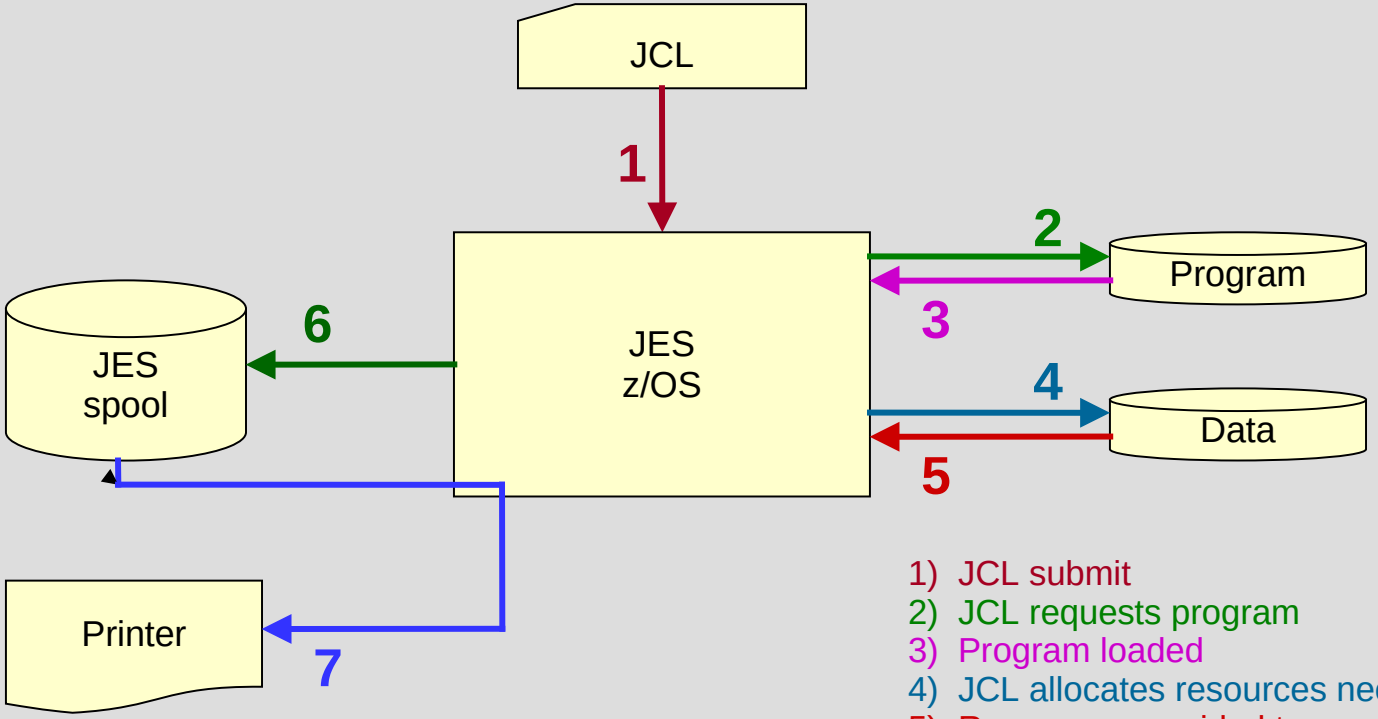
Job Control are the best words describing JCL.

The word "Language" in JCL could easily be replaced by "Syntax" or "Commands" or "Statements".

JCL tells the computer what program to execute.

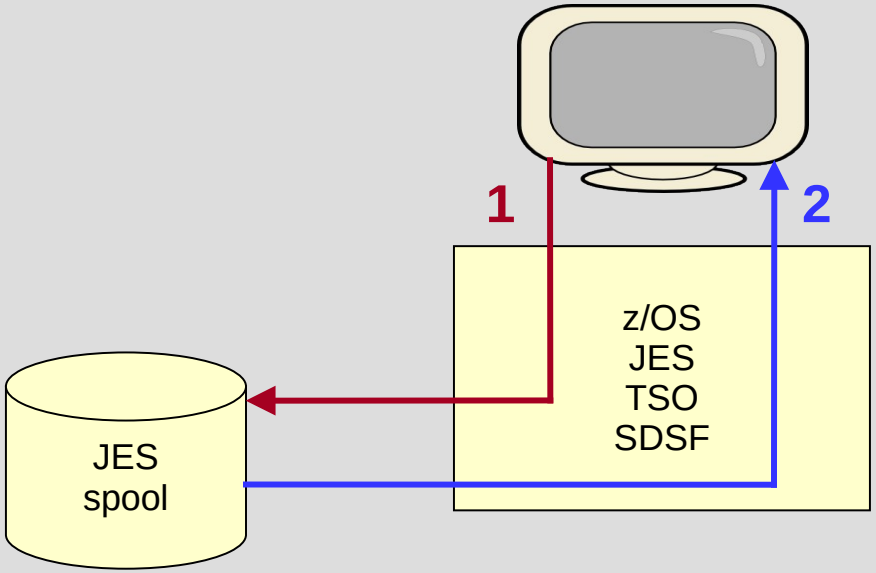
JCL provides a mechanism for the program to read input and write output to requested physical resources.

Job Control Language



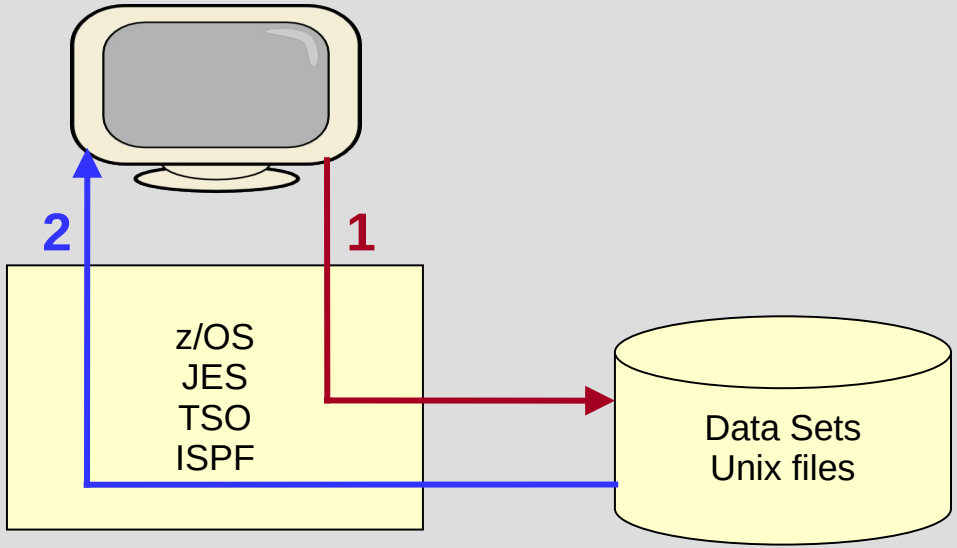
1) JCL submit
2) JCL requests program
3) Program loaded
4) JCL allocates resources needed by program
5) Resources provided to program
6) Program writes output to JES Spool
7) Output to printer as requested

View JCL job output written to JES spool



1) TSO logon using SDSF panels to view JES spool output
2) JES spool output displayed on screen

View JCL job output written to:
MVS Data Sets
Unix files



1) TSO logon using ISPF panels to view program output on disk
2) Data displayed on screen

JCL (**J**ob **C**ontrol **L**anguage)

z/OS written application *programs* include *internal file names* which are *opened* for reading and writing during execution.

The program hard coded file names are only names that are not associated with any physical resources.

JCL *associates* the *program file name* with physical resources such has disk *data set names* or *unix file names*.

JCL is used to process programs in the background (aka 'batch') and to process programs in the foreground (aka 'started task').

JCL submit will result in batch processing of one or more programs.

JCL start will result in foreground processing of processing program.

JCL syntax fundamentals and execution

Job Control Language (JCL) instructs z/OS as a result of "submit" or "start " command.

JCL is easily identified by // in column 1 and 2.

JCL is uppercase unless text is enclosed in quote marks such as unix file names.

Every batch JCL job must contain:

JOB statement
EXEC statement

JOB statement marks the beginning of a batch job and assigns a name to the job.

JCL started tasks do not require a JOB statement

EXEC (execute) statement marks the beginning of a job step, assigns a name to the step, and identifies the program or procedure to be executed in the step.

Every batch job and started task has at least one EXEC statement.

JCL – Job Control Language

Job Control Language (JCL) is a sequential collection of 80 character records beginning with **//** which the operating system reads and interprets

JCL is used to

- Assign name and authority level
- Assign resources (programs, data, etc.) and services needed from the operating system to process a task

JCL can be viewed as a list of statements to be ‘submitted’ for background (batch) processing or ‘started’ for foreground (started task) processing

Minimum JCL batch JOB example:

```
//MYJOB  JOB
//       EXEC    PGM=IEFBR14
```

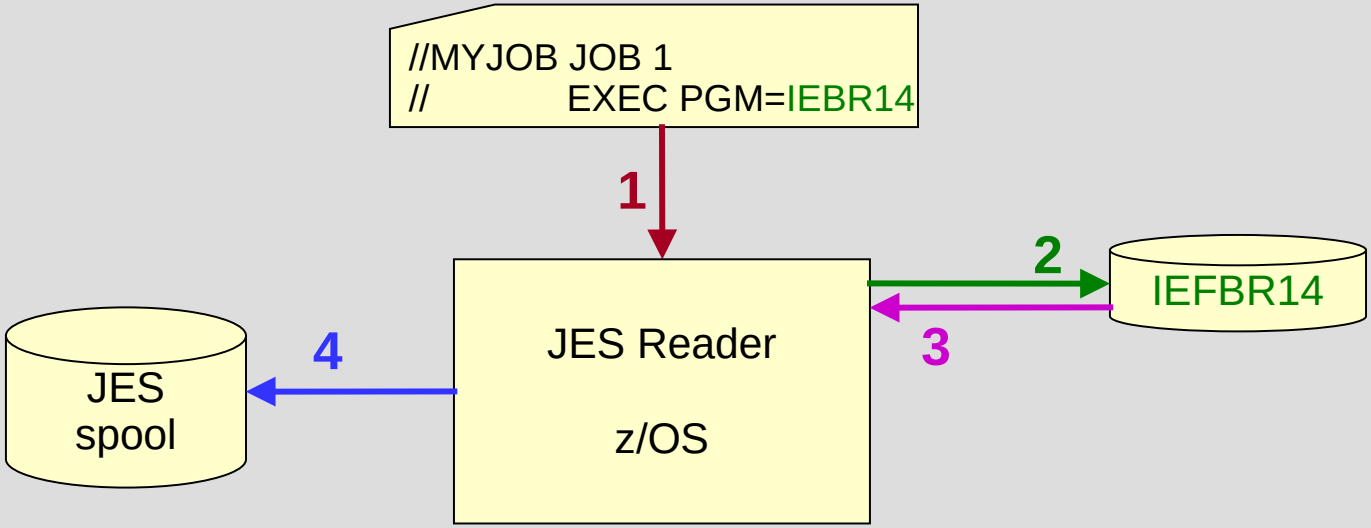
JCL batch job example with stepname of STEP1:

```
//MYJOB  JOB
//STEP1  EXEC    PGM=IEFBR14
```

JCL batch job example with multiple steps:

```
//MYJOB  JOB
//STEP1  EXEC    PGM=IEFBR14
//STEP2  EXEC    PGM=IEFBR14
//STEP3  EXEC    PGM=IEFBR14
```

Job Control Language



1) JCL submit
2) JCL requests program
3) Program loaded
4) Output written to JES spool

JCL DD statements

In addition to the JOB and EXEC statements, jobs may contain one or more DD (Data Definition) statements used to identify and characterize the program input and output.

Example:

```
//MYJOB      JOB
//STEP       EXEC PGM=SORT
//SORTIN     DD   parameters
//SORTOUT    DD   parameters
//SYSIN      DD   parameters
//SYSOUT     DD   parameters
```

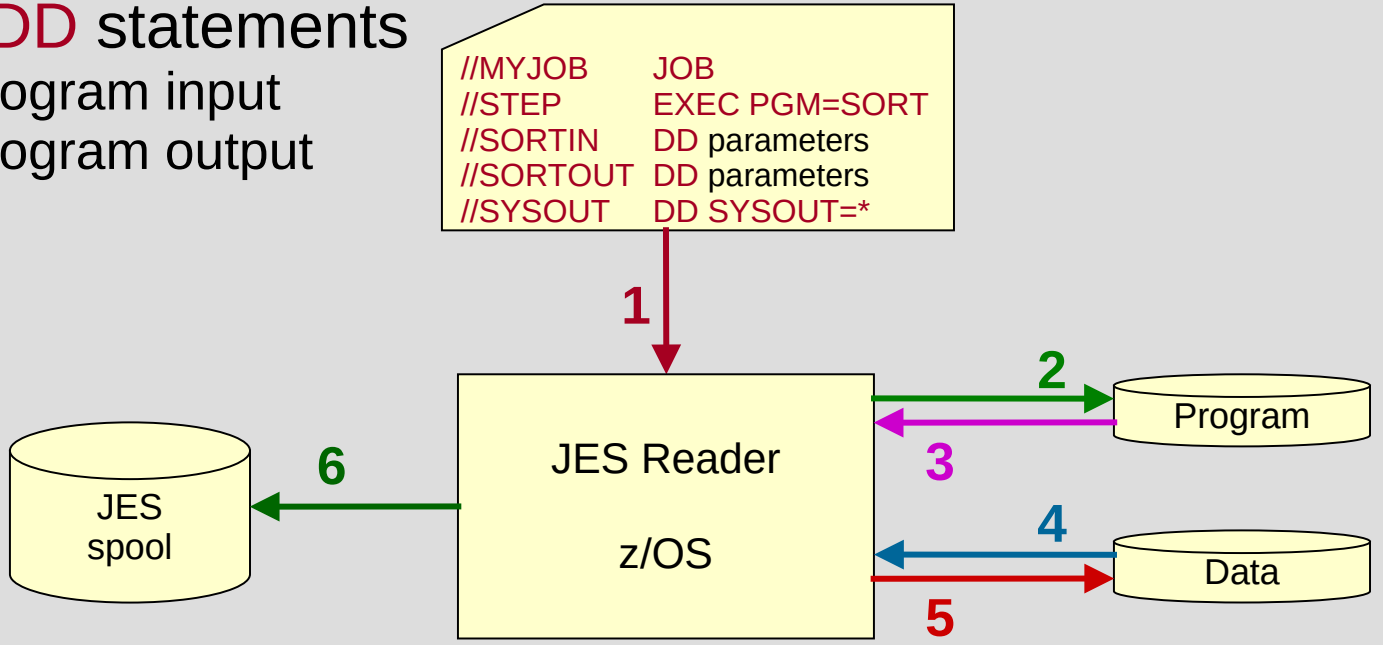
JCL keyword DD is preceded by a 'DD name'.

The above JCL example has 4 'DD names',

SORTIN
SORTOUT
SYSIN
SYSOUT

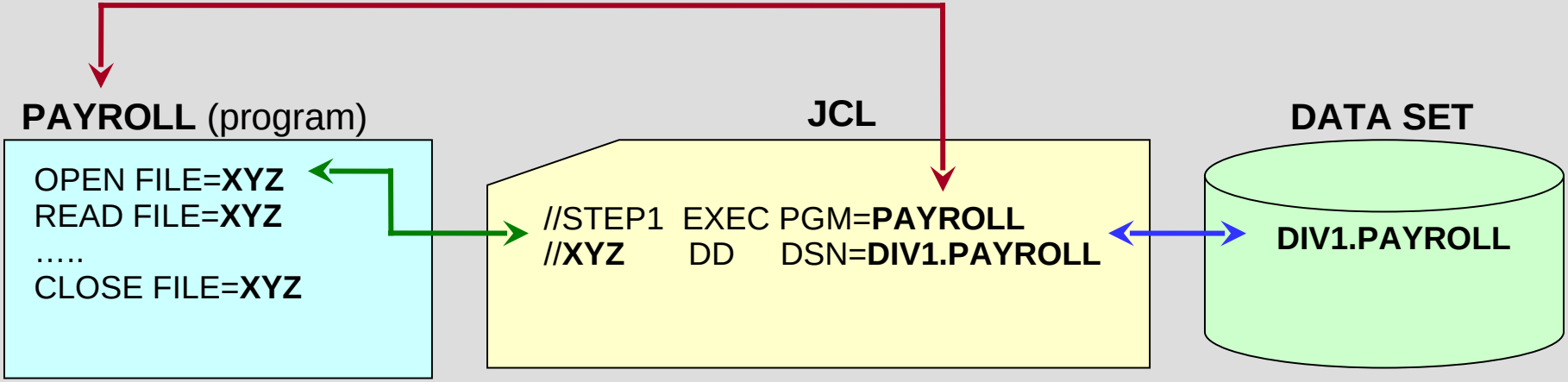
JCL DD statements

program input
program output



1) JCL submit
2) JCL requests program
3) Program loaded
4) JCL //SORTIN DD
5) JCL //SORTOUT DD
6) JCL //SYSOUT DD SYSOUT=*

JCL Referenced DDNAME

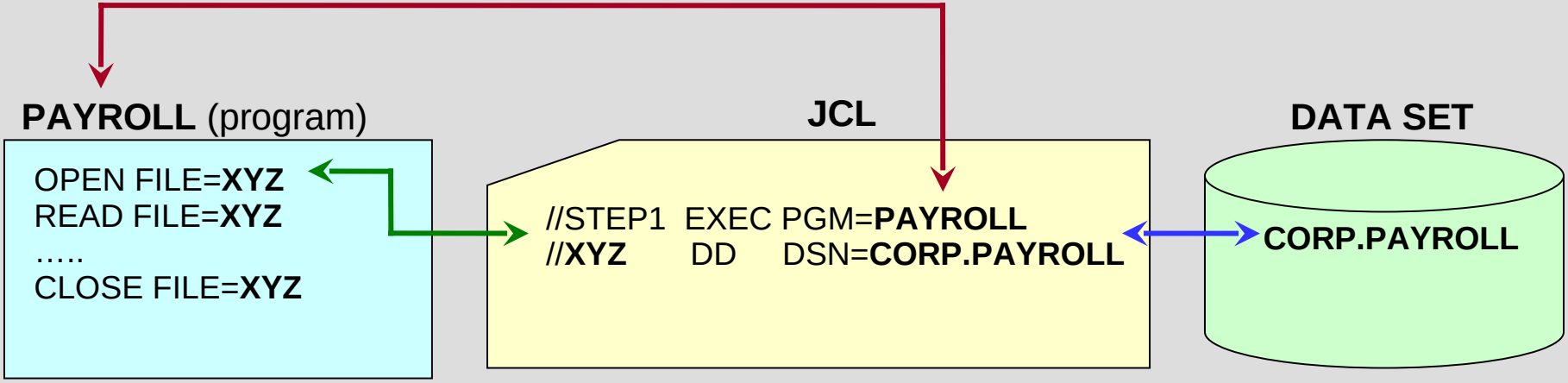


JCL is used to **connect** program **file name** to a z/OS **physical resource** such as a data set name, unix file name, JES spool, printer, network device, etc.

//STEP1 EXEC PGM=PAYROLL *results in open file=xyz*
//XYZ DD DSN=DIV1.PAYROLL *is xyz content read by the program*

DD is abbreviation for **D**ata **D**efinition
XYZ in this example is a program file name
XYZ in this example is also known as the JCL **DDNAME**

JCL Referenced DDNAME



JCL enables ability for same program to read a different z/OS physical resource without changing the program source code

JCL DD statements

DD 'parameters' reference z/OS controlled resources such as unix file name, data set name and data set status

Examples:

PATH='/unixpath/filename'	<<<< unix file name reference
DSN=DATA.SET.NAME	<<<< data set name reference
DISP=(*start*,*end*,*abnormal_end*)	<<<< disposition status of data set

JCL DD DISP=values (*resource disposition*)

```
DISP=(   start      ,end              ,abnormal_end     )
         [NEW]      [,DELETE  ]       [,DELETE  ]
         [OLD]      [,KEEP    ]       [,KEEP    ]
         [SHR]      [,PASS    ]       [,PASS    ]
         [MOD]      [,CATLG   ]       [,CATLG   ]
         [,     ]   [,UNCATLG]       [,UNCATLG]
                    [,        ]
```

OLD	resource exists and exclusive use is requested
SHR	resource exists and may be shared with other requestors
NEW	resource must be created, a new allocation
MOD	data set exists and records to be added at the end, or new data set

DELETE	delete resource when program completes
KEEP	keep resource when program completes
CATLG	update catalog system to locate data set in the future
UNCATLG	update catalog system remove location of resource
PASS	pass the resource to a subsequent JCL step

JCL DD (Data Definition) statements

The program opens DD names as input, output, or both.

The program has an internal file name that will match the JCL DD name.
The association allows different data set names or unix file names to be used by the same program without changing the internal program file name.

When JCL batch job executes, the system writes output to the system controlled JES output queue, data sets and/or unix files as directed by the JCL DD statements

Fundamental JCL statements

The 3 basic JCL statements:

1) JOB statement who wants to process work
2) EXEC statement what program or procedure will be used
3) DD statement what are the program inputs and outputs

Other useful JCL statements:

- PROC and PEND statement execute a JCL predefined procedure
- INCLUDE statement include predefined JCL statements
- IF – THEN – ELSE – ENDIF provides JCL conditional processing

JCL – example

```
//MYJOB         JOB 1
//MYSORT        EXEC   PGM=SORT
//SORTIN        DD     DISP=SHR,DSN=ZIBM000.JCL(AREACODE)
//SORTOUT       DD     SYSOUT=*
//SYSOUT        DD     SYSOUT=*
//SYSIN         DD     *
  SORT FIELDS=(1,3,CH,A)
/*
```

MYJOB is the **jobname**
MYSORT is the **stepname**
SORTIN is program **input**
SORTOUT is program **output**
SYSOUT is system **output** messages
SYSIN is control or data program **input**

JCL - procedures (PROC to PEND)

```
//MYJOB      JOB 1
//MYPROC     PROC
//MYSORT     EXEC PGM=SORT
//SORTIN     DD     DISP=SHR,DSN=&SORTDSN
//SORTOUT    DD     SYSOUT=*
//SYSOUT     DD     SYSOUT=*
//           PEND
```

JCL - procedures (continued)

```
//MYJOB     JOB 1
//*-------------------------------------------------------------------------------------------*
//MYPROC    PROC
//MYSORT    EXEC    PGM=SORT
//SORTIN    DD  DISP=SHR,DSN=&SORTDSN
//SORTOUT   DD  SYSOUT=*
//SYSOUT    DD  SYSOUT=*
//              PEND
//*-------------------------------------------------------------------------------------------*
//STEP1 EXEC    MYPROC,SORTDSN=ZIBM000.JCL(AREACODE)
//SYSIN  DD  *
  SORT FIELDS=(1,3,CH,A)
```

JCL - procedures – PROC statement override

```
//MYJOB         JOB 1
//*-------------------------------------------------------------------------------------------------*
//MYPROC        PROC
//MYSORT        EXEC   PGM=SORT
//SORTIN        DD     DISP=SHR,DSN=&SORTDSN
//SORTOUT       DD     SYSOUT=*
//SYSOUT        DD     SYSOUT=*
//              PEND
//*-------------------------------------------------------------------------------------------------*
//STEP1         EXEC MYPROC,SORTDSN=IBMUSER.AREA.CODES
//MYSORT.SORTOUT   DD DSN=IBMUSER.MYSORT.OUTPUT,DISP=(NEW,CATLG),
//              SPACE=(CYL,(1,1)),UNIT=SYSDA,VOL=SER=SHARED
//SYSIN         DD     *
  SORT FIELDS=(1,3,CH,A)
```

List of Other Commonly Used JCL Operations

```
//name  IF (condition) THEN
//name  ELSE
//name  ENDIF

//name  PROC
//       PEND

//name  SET
```

```
//name  JCLLIB

//name  INCLUDE

//name  COMMAND

//name  OUTPUT

//name  XMIT
```

More exist
JCL grew as needs appeared

Conditional Processing

Test return codes from previous JCL job steps and determine whether to bypass or execute this JCL job step

Old way – will always be available (promise of upward compatibility)

COND=

```
//STEP1  EXEC  PGM=CINDY
  .
  .
//STEP2  EXEC  PGM=NEXT,COND=(4,EQ,STEP1)
```

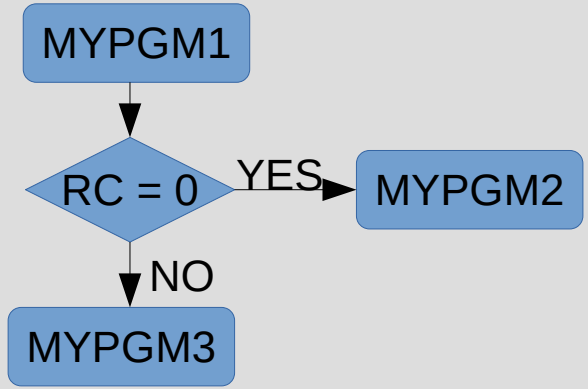
Complex conditional expressions involving multiple previous JCL job steps are possible

Very flexible – but lacked user friendliness

Conditional Processing

New way – IF/THEN/ELSE/ENDIF JCL Operations

```
//START       EXEC PGM=MYPGM1
//    IF RC=0 THEN
//SUCCESS  EXEC PGM=MYPGM2
//    ELSE
//FAILURE     EXEC PGM=MYPGM3
//    ENDIF
```



Complex conditional expressions involving multiple previous JCL job steps are possible

Very flexible – user friendly intent

// IF *condition* THEN / ELSE / ENDIF

Specifies conditional execution of job steps within a job.

```
//TEST     JOB  1
//START    EXEC PGM=IEFBR14
//         IF RC = 0 THEN
//SUCCESS  EXEC PGM=IEFBR14
//         ELSE
//FAILURE  EXEC PGM=IEFBR14
//         ENDIF
```

```
-STEPNAME PROCSTEP    RC   EXCP
-START                00      1
-SUCCESS              00      2
-FAILURE           FLUSH      0
-TEST     ENDED.  NAME-
$HASP395 TEST     ENDED - RC=0000
```

Using system symbols and JCL symbols

Dynamic & Static Symbols

&SYSUID.
&DAY.
&HHMMSS.
&HR.
&JDAY.
&JOBNAME.
&MIN.
&MON.
&SEC.
&WDAY.
&YR2.
&YR4.
&YYMMDD.

&LDAY.
&LHHMMSS.
&LHR.
&LJDAY.
&LMIN.
&LMON.
&LSEC.
&LWDAY.
&LYR2.
&LYR4.
&LYYMMDD.

Using system symbols and JCL symbols

```
//TEST JOB 1,NOTIFY=&SYSUID
//S1   EXEC PGM=IEFBR14
//D1   DD   DSN=&SYSUID..J&JOBNAME..Y&YYMMDD,DISP=(,CATLG),
//          LIKE=&SYSUID..JCL
```

JESJCL Output

```
//TEST JOB 1,NOTIFY=&SYSUID
IEFC653I SUBSTITUTION JCL - 1,NOTIFY=IBMUSER
//S1   EXEC PGM=IEFBR14
//D1   DD   DSN=&SYSUID..J&JOBNAME..Y&YYMMDD,DISP=(,CATLG),
//          LIKE=&SYSUID..JCL
IEFC653I SUBSTITUTION JCL –
DSN=IBMUSER.JJES2.Y180803,DISP=(,CATLG),LIKE=IBMUSER.JCL
```

System symbols and JCL symbols are character strings that represent variable information in JCL.
System symbols allow you to modify JCL statements in a job easily.
A symbol-defining string is limited to eight characters, not including the identifying ampersand (&) character.

** JES2 JOBCLASS SYSSYM=ALLOW

Using System Symbols and JCL Symbols Documentation

Using system symbols and JCL symbols

// SET

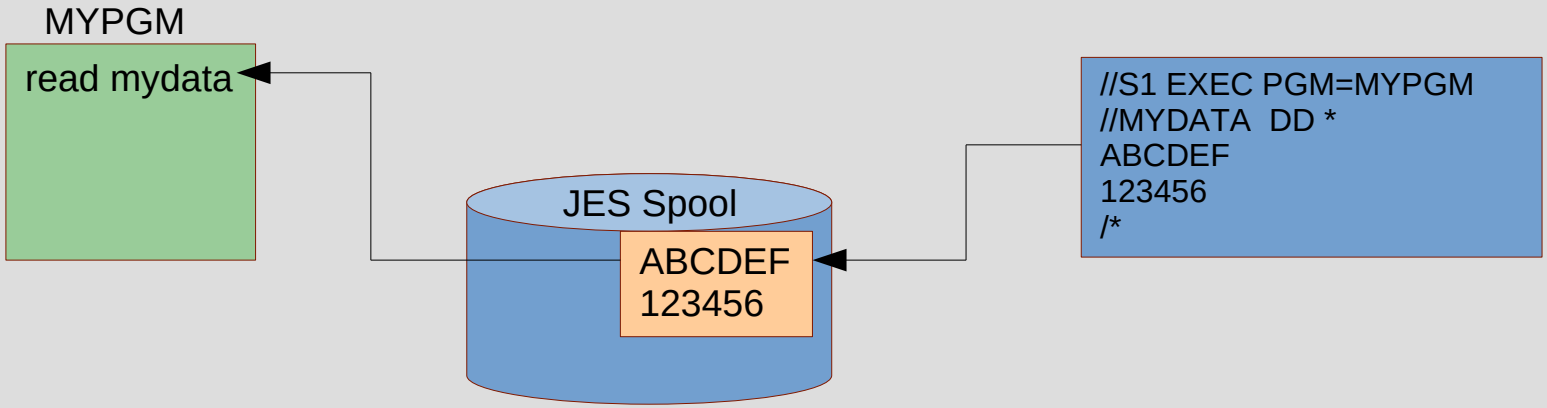
Defines and assigns initial values to symbolic parameters used when processing JCL statements.

Changes or nullifies the values assigned to symbolic parameters.

```
//TEST   JOB   1
//VSET1  SET   P=IEFBR14
//S1     EXEC  PGM=&P
```

JES Enables Very Useful JCL Features

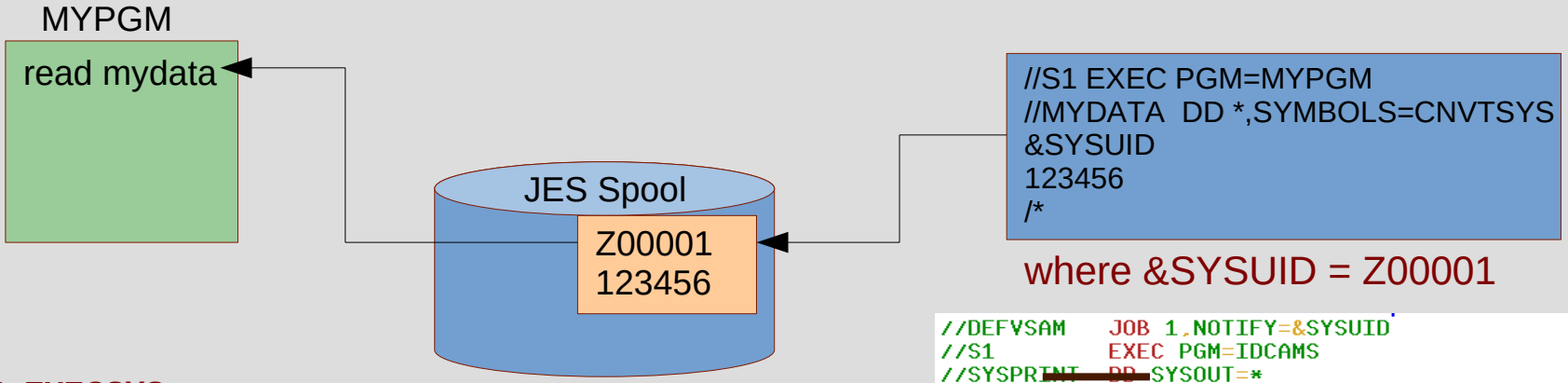
JCL DD * … JES Spool used to store data imbedded in JCL stream



Variables in the
//name DD *
data stream
/*
is possible with JCL DD SYMBOLS parameter

JES Enables Very Useful JCL Features

JCL DD *,**SYMBOLS=** ***enables variable conversion***



SYMBOLS=**EXECSYS**
JCLONLY and system system symbols

SYMBOLS=**CNVTSYS**
EXECSYS and substitute variables on the system where conversion occurred

SYMBOLS=**JCLONLY**
JCL symbols and JES symbols found in the in-stream data set are replaced with their values

```
//DEFVSAM   JOB 1,NOTIFY=&SYSUID
//S1        EXEC PGM=IDCAMS
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *,SYMBOLS=CNVTSYS
  DELETE &SYSUID..VSAM
  SET MAXCC=0
  DEFINE CLUSTER ( NAME ( &SYSUID..VSAM )    -
            VOLUME(VPWRKD) TRACKS(15)        -
            RECORDSIZE(170,170) NONINDEXED   -
            CONTROLINTERVALSIZE(4096) )
//S2        EXEC PGM=IDCAMS
//SYSPRINT  DD SYSOUT=*
//I1        DD DISP=SHR,DSN=&SYSUID..DATA
//O1        DD DISP=SHR,DSN=&SYSUID..VSAM
//SYSIN     DD *
  REPRO INFILE(I1) OUTFILE(O1)
```

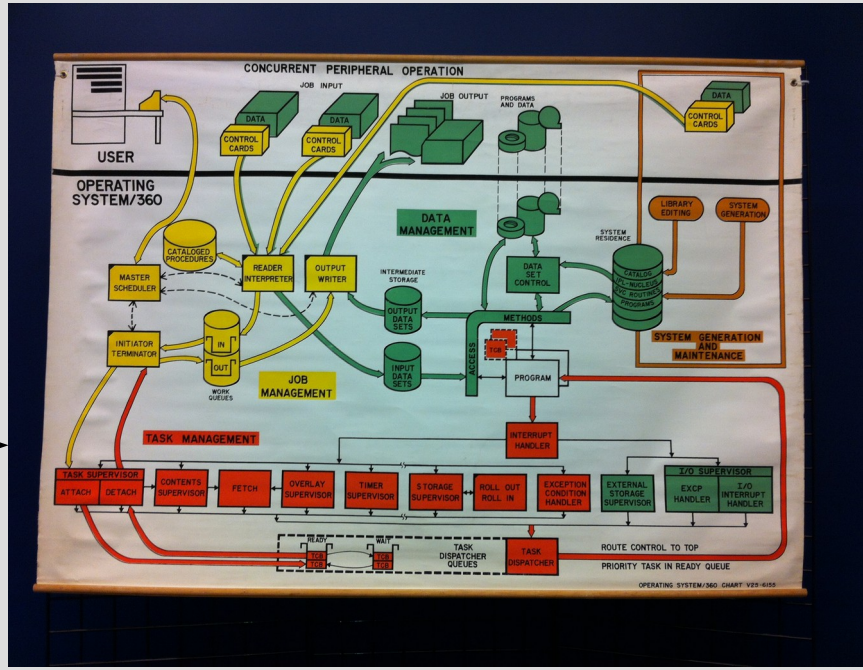
Relationship between JCL and JES

- JES reads and interprets JCL
- JES stores JCL and in-stream data in a JES Spool
- JES collaborates with z/OS to allocate required resources
- JES collects and stores JCL jobname output
- JES itself is JCL ???

Q: So, what reads and interprets the JES JCL Procedure
A: The Master Scheduler

History Lesson – In the beginning JES did not exist

Understanding the master scheduler job control language



View and Understand JCL Job Output Controlled by JES2

JES2 Dynamically Allocates DDNAMEs for each JCL JOBNAME (*JOB*, *STC*, *TSU*)

JESJCLIN
JCL submitted

JESMSGLG
System messages for this job

JESJCL
All job control statements in the input stream

JESYSMSG
JES and operator messages about the job's processing
- allocation of devices and volumes
- execution and termination of job steps and the job
- disposition of data sets

☆ *More about the content in the dynamically allocated DDNAMEs a bit later in this session*

JCL JOB Output Listing

```
SDSF STATUS DISPLAY ALL CLASSES
COMMAND INPUT ===>
NP    JOBNAME  JobID     Owner     Prty Queue
      IBMUSER  TSU00858  IBMUSER     15 EXECUTION
s     TEST     JOB00867  IBMUSER      1 PRINT
? [ ] TEST     JOB00869  IBMUSER      1 PRINT
```

S … select all the JCL JOB output

? .... list all the JCL JOB DDNAMEs

JCL JOB Dynamically Allocated DDNAMEs

```
SDSF JOB DATA SET DISPLAY - JOB TEST     (JOB00867)
COMMAND INPUT ===>
NP    DDNAME   StepName ProcStep DSID Owner     C Dest
s     JESJCLIN                    1 IBMUSER   W
s     JESMSGLG JES2               2 IBMUSER   W LOCAL
s     JESJCL   JES2               3 IBMUSER   W LOCAL
s     JESYSMSG JES2               4 IBMUSER   W LOCAL
```

JESJCLIN Output .. w/JCL error

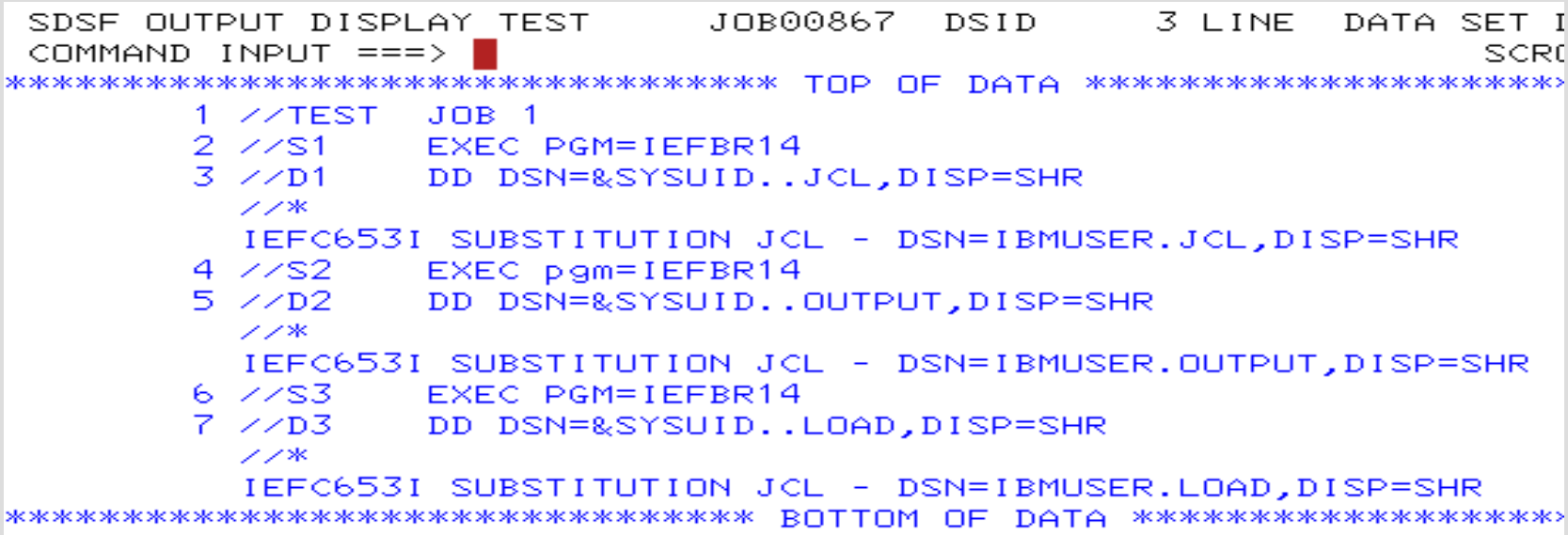
```
 SDSF OUTPUT DISPLAY TEST      JOB00867  DSID
 COMMAND INPUT ===>
****************************************** TOP OF DATA ***
//TEST   JOB 1
//S1     EXEC PGM=IEFBR14
//D1     DD DSN=&SYSUID..JCL,DISP=SHR
//*
//S2     EXEC pgm=IEFBR14
//D2     DD DSN=&SYSUID..OUTPUT,DISP=SHR
//*
//S3     EXEC PGM=IEFBR14
//D3     DD DSN=&SYSUID..LOAD,DISP=SHR
//*
****************************************** BOTTOM OF DATA *
```

JESMSGLG Output .. w/JCL error (continued)

```
 SDSF OUTPUT DISPLAY TEST     JOB00867  DSID     2 LINE  DATA SET DIS
 COMMAND INPUT ===>                                          SCROLL
****************************************** TOP OF DATA ***********************
                    J E S 2  J O B  L O G  --  S Y S T E M  S 0 W 1  -

08.28.27 JOB00867 ---- SUNDAY,     24 JUN 2018 ----
08.28.27 JOB00867  IRR010I  USERID IBMUSER  IS ASSIGNED TO THIS JOB.
08.28.27 JOB00867  IEFC452I TEST - JOB NOT RUN - JCL ERROR  530
------ JES2 JOB STATISTICS ------
            10 CARDS READ
            29 SYSOUT PRINT RECORDS
             0 SYSOUT PUNCH RECORDS
             1 SYSOUT SPOOL KBYTES
          0.00 MINUTES EXECUTION TIME
****************************************** BOTTOM OF DATA ********************
```

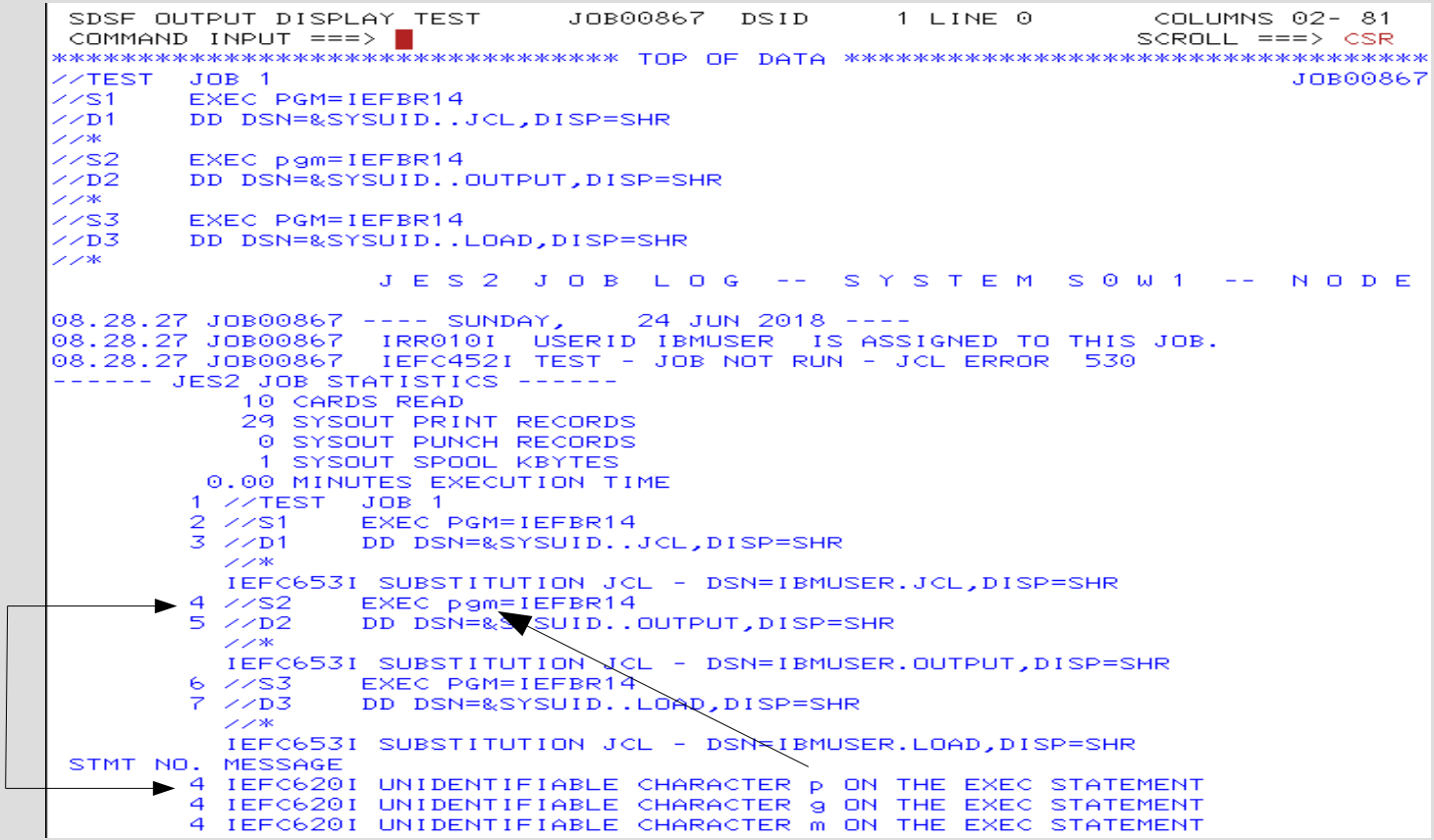
JESJCL Output .. w/JCL error (continued)



JESYSMSG Output .. w/JCL error (continued)

```
SDSF OUTPUT DISPLAY TEST     JOB00867  DSID     4 LINE  DATA SET DIS
COMMAND INPUT ===>                                          SCROLL
****************************** TOP OF DATA ***********************
STMT NO. MESSAGE
       4 IEFC620I UNIDENTIFIABLE CHARACTER p ON THE EXEC STATEMENT
       4 IEFC620I UNIDENTIFIABLE CHARACTER g ON THE EXEC STATEMENT
       4 IEFC620I UNIDENTIFIABLE CHARACTER m ON THE EXEC STATEMENT
****************************** BOTTOM OF DATA ********************
```

JCL JOB Output .. w/JCL error (continued)



```
 SDSF OUTPUT DISPLAY TEST      JOB00867  DSID      1 LINE 0        COLUMNS 02- 81
 COMMAND INPUT ===>                                              SCROLL ===> CSR
****************************** TOP OF DATA ********************************
//TEST   JOB 1                                                      JOB00867
//S1     EXEC PGM=IEFBR14
//D1     DD DSN=&SYSUID..JCL,DISP=SHR
//*
//S2     EXEC pgm=IEFBR14
//D2     DD DSN=&SYSUID..OUTPUT,DISP=SHR
//*
//S3     EXEC PGM=IEFBR14
//D3     DD DSN=&SYSUID..LOAD,DISP=SHR
//*
                    J E S 2  J O B  L O G  --  S Y S T E M  S 0 W 1  --  N O D E

08.28.27 JOB00867 ---- SUNDAY,     24 JUN 2018 ----
08.28.27 JOB00867  IRR010I  USERID IBMUSER  IS ASSIGNED TO THIS JOB.
08.28.27 JOB00867  IEFC452I TEST - JOB NOT RUN - JCL ERROR  530
------ JES2 JOB STATISTICS ------
            10 CARDS READ
            29 SYSOUT PRINT RECORDS
             0 SYSOUT PUNCH RECORDS
             1 SYSOUT SPOOL KBYTES
          0.00 MINUTES EXECUTION TIME
         1 //TEST   JOB 1
         2 //S1     EXEC PGM=IEFBR14
         3 //D1     DD DSN=&SYSUID..JCL,DISP=SHR
           //*
           IEFC653I SUBSTITUTION JCL - DSN=IBMUSER.JCL,DISP=SHR
         4 //S2     EXEC pgm=IEFBR14
         5 //D2     DD DSN=&SYSUID..OUTPUT,DISP=SHR
           //*
           IEFC653I SUBSTITUTION JCL - DSN=IBMUSER.OUTPUT,DISP=SHR
         6 //S3     EXEC PGM=IEFBR14
         7 //D3     DD DSN=&SYSUID..LOAD,DISP=SHR
           //*
           IEFC653I SUBSTITUTION JCL - DSN=IBMUSER.LOAD,DISP=SHR
 STMT NO.  MESSAGE
         4  IEFC620I UNIDENTIFIABLE CHARACTER p ON THE EXEC STATEMENT
         4  IEFC620I UNIDENTIFIABLE CHARACTER g ON THE EXEC STATEMENT
         4  IEFC620I UNIDENTIFIABLE CHARACTER m ON THE EXEC STATEMENT
```

JESJCLIN Output (continued)

```
 SDSF OUTPUT DISPLAY TEST      JOB00869  DSID
 COMMAND INPUT ===>
********************************** TOP OF DATA ***>
//TEST   JOB 1
//S1     EXEC PGM=IEFBR14
//D1     DD DSN=&SYSUID..JCL,DISP=SHR
//*
//S2     EXEC PGM=IEFBR14
//D2     DD DSN=&SYSUID..OUTPUT,DISP=SHR
//*
//S3     EXEC PGM=IEFBR14
//D3     DD DSN=&SYSUID..LOAD,DISP=SHR
//*
******************************** BOTTOM OF DATA *>
```

JCL error correction

JESMSGLG Output (continued)

```
 SDSF OUTPUT DISPLAY TEST     JOB00869  DSID     2 LINE  DATA SET DISPLAYED
 COMMAND INPUT ===>                                        SCROLL ===> CSR
****************************************** TOP OF DATA ******************************************
                     J E S 2  J O B  L O G  --  S Y S T E M  S 0 W 1  --  N O D E

08.38.36 JOB00869 ---- SUNDAY,     24 JUN 2018 ----
08.38.36 JOB00869  IRR010I  USERID IBMUSER  IS ASSIGNED TO THIS JOB.
08.38.36 JOB00869  ICH70001I IBMUSER  LAST ACCESS AT 08:27:48 ON SUNDAY, JUNE 24
08.38.36 JOB00869  $HASP373 TEST     STARTED - INIT 1     - CLASS A         - SYS
08.38.36 JOB00869  -                                          -----TIMINGS (MINS.)--
08.38.36 JOB00869  -STEPNAME PROCSTEP    RC   EXCP    CONN       TCB       SRB   C
08.38.36 JOB00869  -S1                   00      1       0       .00       .00
08.38.36 JOB00869  -S2                   00      2       0       .00       .00
08.38.37 JOB00869  -S3                   00      2       0       .00       .00
08.38.37 JOB00869  -TEST     ENDED.  NAME-                     TOTAL TCB CPU TIM
08.38.37 JOB00869  $HASP395 TEST     ENDED - RC=0000
------ JES2 JOB STATISTICS ------
  24 JUN 2018 JOB EXECUTION DATE
           10 CARDS READ
           78 SYSOUT PRINT RECORDS
            0 SYSOUT PUNCH RECORDS
           10 SYSOUT SPOOL KBYTES
         0.00 MINUTES EXECUTION TIME
****************************************** BOTTOM OF DATA ******************************************
```

JESJCL Output (continued)

```
 SDSF OUTPUT DISPLAY TEST      JOB00869  DSID      3 LINE  DATA SET DISP
 COMMAND INPUT ===>                                              SCROLL
****************************************** TOP OF DATA ***********************
         1 //TEST  JOB 1
         2 //S1    EXEC PGM=IEFBR14
         3 //D1    DD DSN=&SYSUID..JCL,DISP=SHR
           //*
           IEFC653I SUBSTITUTION JCL - DSN=IBMUSER.JCL,DISP=SHR
         4 //S2    EXEC PGM=IEFBR14
         5 //D2    DD DSN=&SYSUID..OUTPUT,DISP=SHR
           //*
           IEFC653I SUBSTITUTION JCL - DSN=IBMUSER.OUTPUT,DISP=SHR
         6 //S3    EXEC PGM=IEFBR14
         7 //D3    DD DSN=&SYSUID..LOAD,DISP=SHR
           //*
           IEFC653I SUBSTITUTION JCL - DSN=IBMUSER.LOAD,DISP=SHR
****************************************** BOTTOM OF DATA ********************
```

JESYSMSG Output (continued)

```
 SDSF OUTPUT DISPLAY TEST     JOB00869  DSID     4 LINE  DATA SET DISPLAYED
 COMMAND INPUT ===>                                          SCROLL ===> CSR
****************************************** TOP OF DATA ******************************************
ICH70001I IBMUSER  LAST ACCESS AT 08:27:48 ON SUNDAY, JUNE 24, 2018
IEFA111I TEST IS USING THE FOLLOWING JOB RELATED SETTINGS:
         SWA=ABOVE,TIOT SIZE=32K,DSENQSHR=DISALLOW,GDGBIAS=JOB
IEF236I ALLOC. FOR TEST S1
IGD103I SMS ALLOCATED TO DDNAME D1
IEF142I TEST S1 - STEP WAS EXECUTED - COND CODE 0000
IGD104I IBMUSER.JCL                                  RETAINED,  DDNAME=D1
IEF373I STEP/S1       /START 2018175.0838
IEF032I STEP/S1       /STOP  2018175.0838
         CPU:       0 HR  00 MIN  00.00 SEC    SRB:       0 HR  00 MIN  00.00 SEC
         VIRT:       4K  SYS:    228K  EXT:          0K  SYS:     10888K
         ATB- REAL:                      12K  SLOTS:                    0K
              VIRT- ALLOC:        10M SHRD:         0M
IEF236I ALLOC. FOR TEST S2
IEF237I 0D31 ALLOCATED TO D2
IEF142I TEST S2 - STEP WAS EXECUTED - COND CODE 0000
IEF285I    IBMUSER.OUTPUT                             KEPT
IEF285I    VOL SER NOS= VPWRKB.
IEF373I STEP/S2       /START 2018175.0838
IEF032I STEP/S2       /STOP  2018175.0838
         CPU:       0 HR  00 MIN  00.00 SEC    SRB:       0 HR  00 MIN  00.00 SEC
         VIRT:       4K  SYS:    228K  EXT:          0K  SYS:     10884K
         ATB- REAL:                      12K  SLOTS:                    0K
              VIRT- ALLOC:        10M SHRD:         0M
IEF236I ALLOC. FOR TEST S3
IGD103I SMS ALLOCATED TO DDNAME D3
IEF142I TEST S3 - STEP WAS EXECUTED - COND CODE 0000
IGD104I IBMUSER.LOAD                                 RETAINED,  DDNAME=D3
IEF373I STEP/S3       /START 2018175.0838
IEF032I STEP/S3       /STOP  2018175.0838
         CPU:       0 HR  00 MIN  00.00 SEC    SRB:       0 HR  00 MIN  00.00 SEC
         VIRT:       4K  SYS:    228K  EXT:          0K  SYS:     10884K
         ATB- REAL:                      12K  SLOTS:                    0K
              VIRT- ALLOC:        10M SHRD:         0M
IEF375I  JOB/TEST     /START 2018175.0838
IEF033I  JOB/TEST     /STOP  2018175.0838
         CPU:       0 HR  00 MIN  00.00 SEC    SRB:       0 HR  00 MIN  00.00 SEC
****************************************** BOTTOM OF DATA ***************************************
```

JCL Procedures

//name **PROC**

Marks the **beginning** of either

1. in-stream procedure
2. cataloged procedure

assigns values to parameters defined in the procedure

//name **PEND**

Marks the **end** of either

1. in-stream procedure
2. cataloged procedure

JCL Procedures (**In-Stream** with parameter value substitution)

```
//TEST    JOB  1
//*-------------------------
//MYPROC  PROC P=
//PSTEP1  EXEC PGM=&P
//       PEND
//*-------------------------
//MYJCL   EXEC MYPROC,P=IEFBR14
```

P is assigned as a PROC variable

P is assigned value **IEFBR14**

```
1 //TEST    JOB  1
  //*-------------------------
2 //MYPROC  PROC P=
  //PSTEP1  EXEC PGM=&P
  //       PEND
  //*-------------------------
3 //MYJCL   EXEC MYPROC,P=IEFBR14
4 ++MYPROC  PROC P=
5 ++PSTEP1  EXEC PGM=&P
  IEFC653I SUBSTITUTION JCL - PGM=IEFBR14
```

Observe lines **4** & **5**
++ JCL In-Stream
Procedure Expanded Statements

JESJCL Output for In-Stream Procedures

++ …. DD statement that was not overridden and all other JCL statements, except the JCL comment statement. Each statement appears in the listing exactly as it appears in the procedure.

+/ …. DD statement that was overridden (preceded by the overriding DD statement)

++* ….Job control statement that is not a JCL comment statement but one that the system considers to contain only comments

++* …. JCL comment statement

JCL Procedures (**Cataloged** Procedure)

```
//TEST    JOB  1
//MYJCL   EXEC MYPROC,P=IEFBR14
```

```
1 //TEST    JOB  1
2 //MYJCL   EXEC MYPROC,P=IEFBR14
3 XXMYPROC PROC P=
4 XXPSTEP1 EXEC PGM=&P
  IEFC653I SUBSTITUTION JCL - PGM=IEFBR14
5 XX       PEND
```

Observe lines **3**, **4** & **5**
XX JCL Cataloged Procedure Expanded Statements

JESJCL Output for Cataloged Procedures

XX …. DD statement that was not overridden and all other JCL statements,except the JCL comment statement. Each statement appears in the listing exactly as it appears in the procedure

X/ …. DD statement that was overridden (preceded by the overriding DD statement)

XX* … Job control statement that is not a JCL comment statement but one that the system considers to contain only comments

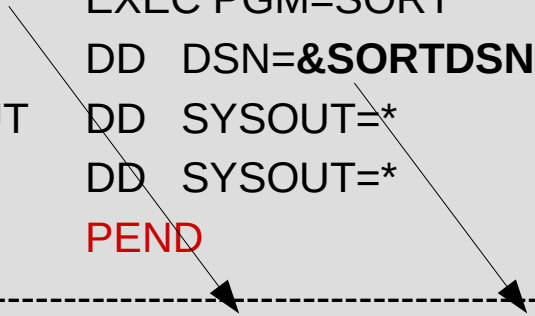
XX* …. JCL comment statement

JCL Procedures (PROC to PEND)

```
//MYJOB      JOB 1
//MYPROC     PROC
//MYSORT     EXEC PGM=SORT
//SORTIN     DD   DSN=&SORTDSN,DISP=SHR
//SORTOUT    DD   SYSOUT=*
//SYSOUT     DD   SYSOUT=*
//           PEND
```

JCL Procedures

```
//MYJOB       JOB 1
//*-------------------------------------------------------------------------------------*
//MYPROC      PROC
//MYSORT      EXEC PGM=SORT
//SORTIN      DD   DSN=&SORTDSN,DISP=SHR
//SORTOUT     DD   SYSOUT=*
//SYSOUT      DD   SYSOUT=*
//             PEND
//*-------------------------------------------------------------------------------------*
//STEP1       EXEC MYPROC,SORTDSN=ZIBM000.JCL(AREACODE)
//SYSIN       DD   *
  SORT FIELDS=(1,3,CH,A)
```



JCL Procedures – Statement Override

//STEPNAME.DDNAME DD ….

```
//MYJOB        JOB 1
//*
//MYPROC       PROC
//MYSORT       EXEC   PGM=SORT
//SORTIN       DD     DSN=&SORTDSN,DISP=SHR
//SORTOUT      DD     SYSOUT=*
//SYSOUT       DD     SYSOUT=*
//                    PEND
//*
//STEP1        EXEC MYPROC,SORTDSN=IBMUSER.AREA.CODES
//MYSORT.SORTOUT  DD DSN=IBMUSER.MYSORT.OUTPUT,
//       DISP=(NEW,CATLG),
//       SPACE=(CYL,(1,1)),
//       UNIT=SYSDA,VOL=SER=VPWRKA
//SYSIN        DD  *
  SORT FIELDS=(1,3,CH,A)
```

In-stream JCL Procedure w/Override (JESJCLIN)

```
 SDSF OUTPUT DISPLAY SORTJOB  JOB00874  DSID     1 LINE 0
 COMMAND INPUT ===>
********************************* TOP OF DATA ***********
//SORTJOB JOB 1,NOTIFY=&SYSUID
//*-----------------------------*/
//MYPROC  PROC
//MYSORT  EXEC PGM=SORT
//SYSOUT   DD SYSOUT=*
//SORTOUT  DD SYSOUT=*
//SORTIN   DD DISP=SHR,DSN=&SORTDSN
//        PEND
//*-----------------------------*/
//STEP1   EXEC MYPROC,SORTDSN=CLASS.LAB.JCL(AREACODE)
//MYSORT.SORTOUT DD DSN=&SYSUID..SORT.OUTPUT,
//           DISP=(NEW,CATLG),SPACE=(CYL,(1,1)),UNIT=SYSDA,
//           DCB=(LRECL=20,BLKSIZE=0,RECFM=FB,DSORG=PS)
//SYSIN    DD *
******************************** BOTTOM OF DATA ***********
```

In-stream JCL Procedure w/Override (JESMSGLG)

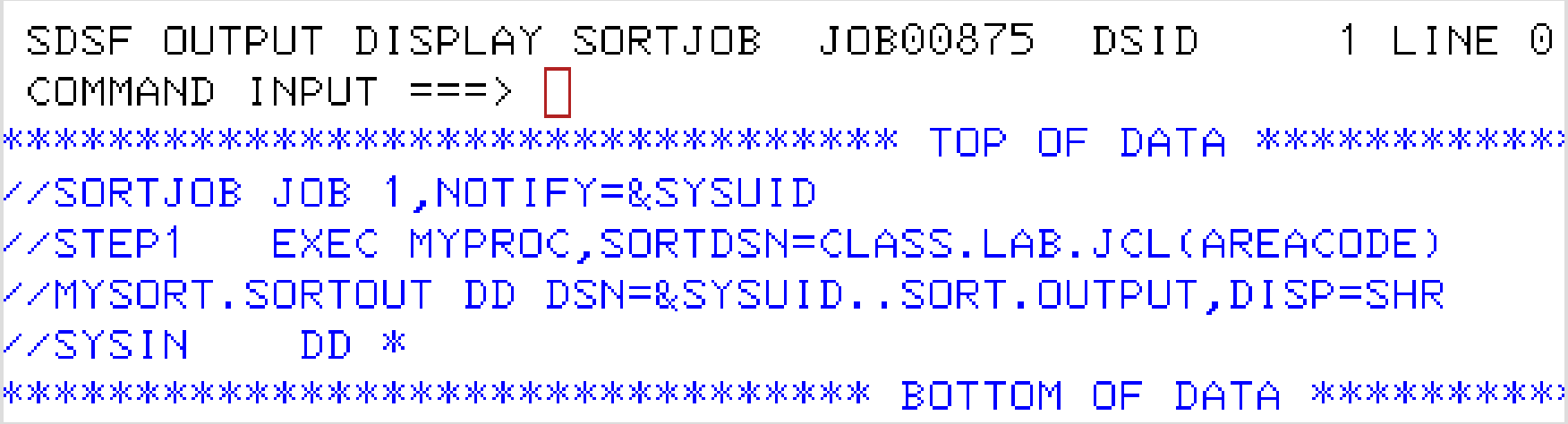
```
 SDSF OUTPUT DISPLAY SORTJOB  JOB00874  DSID     2 LINE  DATA SET DISPLAYED
 COMMAND INPUT ===>                                        SCROLL ===> CSR
****************************** TOP OF DATA ******************************
                  J E S 2  J O B  L O G  --  S Y S T E M  S 0 W 1  --  N O D E

10.11.07 JOB00874 ---- SUNDAY,    24 JUN 2018 ----
10.11.07 JOB00874  IRR010I  USERID IBMUSER  IS ASSIGNED TO THIS JOB.
10.11.07 JOB00874  ICH70001I IBMUSER  LAST ACCESS AT 08:38:36 ON SUNDAY, JUNE 24
10.11.07 JOB00874  $HASP373 SORTJOB  STARTED - INIT 1    - CLASS A         - SYS
10.11.08 JOB00874  -                                    -----TIMINGS (MINS.)--
10.11.08 JOB00874  -STEPNAME PROCSTEP    RC   EXCP   CONN       TCB       SRB  C
10.11.08 JOB00874  -STEP1    MYSORT      00     43      3       .00       .00
10.11.08 JOB00874  -SORTJOB  ENDED.  NAME-                   TOTAL TCB CPU TIM
10.11.08 JOB00874  $HASP395 SORTJOB  ENDED - RC=0000
------ JES2 JOB STATISTICS ------
  24 JUN 2018 JOB EXECUTION DATE
           16 CARDS READ
          111 SYSOUT PRINT RECORDS
            0 SYSOUT PUNCH RECORDS
           10 SYSOUT SPOOL KBYTES
         0.01 MINUTES EXECUTION TIME
****************************** BOTTOM OF DATA ******************************
```

In-stream JCL Procedure w/Override (JESJCL)

```
SDSF OUTPUT DISPLAY SORTJOB  JOB00874  DSID     3 LINE  DATA SET DISPLAYED
COMMAND INPUT ===>                                          SCROLL ===> CSR
****************************************** TOP OF DATA ******************************************
        1 //SORTJOB JOB 1,NOTIFY=&SYSUID
          //*-------------------------*/
          IEFC653I SUBSTITUTION JCL - 1,NOTIFY=IBMUSER
        2 //MYPROC  PROC
          //MYSORT  EXEC PGM=SORT
          //SYSOUT   DD SYSOUT=*
          //SORTOUT  DD SYSOUT=*
          //SORTIN   DD DISP=SHR,DSN=&SORTDSN
          //         PEND
          //*-------------------------*/
        3 //STEP1   EXEC MYPROC,SORTDSN=CLASS.LAB.JCL(AREACODE)
        4 ++MYPROC  PROC
        5 ++MYSORT  EXEC PGM=SORT
        6 ++SYSOUT   DD SYSOUT=*
        7 //MYSORT.SORTOUT DD DSN=&SYSUID..SORT.OUTPUT,
          //           DISP=(NEW,CATLG),SPACE=(CYL,(1,1)),UNIT=SYSDA,
          //           DCB=(LRECL=20,BLKSIZE=0,RECFM=FB,DSORG=PS)
          IEFC653I SUBSTITUTION JCL - DSN=IBMUSER.SORT.OUTPUT,DISP=(NEW,CATLG),S
          DCB=(LRECL=20,BLKSIZE=0,RECFM=FB,DSORG=PS)
          +/SORTOUT  DD SYSOUT=*
        8 ++SORTIN   DD DISP=SHR,DSN=&SORTDSN
          IEFC653I SUBSTITUTION JCL - DISP=SHR,DSN=CLASS.LAB.JCL(AREACODE)
        9 //SYSIN    DD *
****************************************** BOTTOM OF DATA ***************************************
```

Cataloged JCL Procedure (JESJCLIN)



Cataloged JCL Procedure (JESJCL)

```
 SDSF OUTPUT DISPLAY SORTJOB  JOB00875  DSID     3 LINE  DATA SET DISPLAYED
 COMMAND INPUT ===>                                          SCROLL ===> CSR
****************************** TOP OF DATA ******************************
        1 //SORTJOB JOB 1,NOTIFY=&SYSUID
          IEFC653I SUBSTITUTION JCL - 1,NOTIFY=IBMUSER
        2 //STEP1   EXEC MYPROC,SORTDSN=CLASS.LAB.JCL(AREACODE)
        3 XXMYPROC  PROC
        4 XXMYSORT  EXEC PGM=SORT
        5 XXSYSOUT   DD SYSOUT=*
        6 XXSORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(5,1))
        7 //MYSORT.SORTOUT DD DSN=&SYSUID..SORT.OUTPUT,DISP=SHR
          IEFC653I SUBSTITUTION JCL - DSN=IBMUSER.SORT.OUTPUT,DISP=SHR
          X/SORTOUT  DD SYSOUT=*
        8 XXSORTIN   DD DISP=SHR,DSN=&SORTDSN
          IEFC653I SUBSTITUTION JCL - DISP=SHR,DSN=CLASS.LAB.JCL(AREACODE)
        9 //SYSIN    DD *
       10 XX         PEND
****************************** BOTTOM OF DATA ***************************
```

Create and Pass Temporary Data Set Between JCL STEPs

Temporary Data Sets

A temporary data set is a data set that is created and deleted in the same job, and is identified by coding one of the following:

```
//TEST  JOB 1,NOTIFY=&SYSUID
//*-------------------------
//STEP1 EXEC PGM=IEFBR14
//D1    DD   DSN=&&TMP,DISP=(NEW,PASS),SPACE=(TRK,1)
//*-------------------------
//STEP2 EXEC PGM=IEFBR14
//D1    DD   DSN=&&TMP,DISP=(OLD,DELETE)
```

DSNAME=&&dsname

For a temporary data set

DSNAME=&&dsname(member)

For a member of a temporary PDS or PDSE

No DSNAME parameter

For a temporary data set to be named by the system

Miscellaneous JCL Operations

//*name* **SET**

Defines and assigns values to symbolic parameters
used when processing JCL statements.

//name **JCLLIB** ORDER=(*names of the libraries to be searched*)

```
//SET1          SET LIB=MY.JCLLIB,D=MY.INPUT.DATA,M=AA
//*
//PRIVATE       JCLLIB ORDER=(&LIB)
//*
//*   search for MYPROC first in MY.JCLLIB
//*
//COPY          EXEC MYPROC
//INDATA        DD     DSN=&D,DISP=SHR
//MOREJCL   INCLUDE MEMBER=&M
```

// JCLLIB
// INCLUDE

Identifies the **libraries** that the system will **search for**:

- Procedures named in EXEC statements
- INCLUDE groups

```
//TEST    JOB  1
//MYLIBS JCLLIB ORDER=(IBMUSER.JCL)
//GO      EXEC TESTPROC
//GETJCL INCLUDE MEMBER=DD1
```

```
1 //TEST    JOB  1
2 //MYLIBS JCLLIB ORDER=(IBMUSER.JCL)
3 //GO      EXEC TESTPROC
4 XXTESTPROC PROC
5 XXPSTEP1   EXEC PGM=IEFBR14
6 XXDUMMY1  DD DUMMY
7 XXDUMMY2  DD DUMMY
8 XX          PEND
9 //GETJCL INCLUDE MEMBER=DD1
```

JOB Operation Parameters

TYPRUN=	
SCAN	check JCL syntax
HOLD	hold until command to release
JCLHOLD	JES2 hold until command to release
COPY	copy JCL to output without processing
NOTIFY=	
&SYSUID	any valid ID
TIME=	modify default processing time
REGION=	modify default processing memory
MEMLIMIT=	
PAGES=	modify default output volume
LINES=	
EMAIL=	

Many more

DD Operation DCB= parameter

Used to assign attributes to a resource such as a data set name

- Logical Record Length
- Record Format
- Data Set Organization

DCB, Data Control Block, operands

- LRECL=
- RECFM=
- DSORG=

Assembler Macro

LIKE= parameter exists for newly allocated data set names

SMS, Storage Management Subsystem

SMS ACS Routine Impact on DD Operation

DSN=
MGMTCLAS=
STORCLAS=
DATACLAS=

SMS Enables Disk Storage Administrators to simplify JCL DD parameters

SMS Locally documented JCL procedures and policy

SMS ACS, Automatic Class Selection, routine parses and changes JCL
Routine discards user JCL DD operands and substitutes different JCL DD parameters

** interally generated DD operation defined physical resources
** what you code as a DD operation and submit can be altered by SMS ACS routine

System Utilities & Old Tricks

SORT & ICETOOL	powerful and flexible data sorting, filtering, and field manipulation
IKJEFT01	anything possible from interactive TSO can be processed using JCL
IDCAMS	create, delete, rename, copy data for VSAM and non-VSAM
IEBCOPY	copy PDS members
IEBGENER	copy sequential data
IEBDG	data generator
IEFBR14	dummy program useful for allocating and deleting data sets
BPXBATCH	Unix utility to process Unix shell commands or programs using JCL
– many more	learn the utilities - “don’t write programs when utility will do the job”

JCL can be used to submit another JCL JOB
    //RDR DD SYSOUT=(,INTRDR)

REXX can be used to build and submit JCL JOB

FTP can used to submit JCL from workstation, etc.
    FTP can used to retrieve the JES output

z/OS Internet Library >

z/OS V2R4 Library

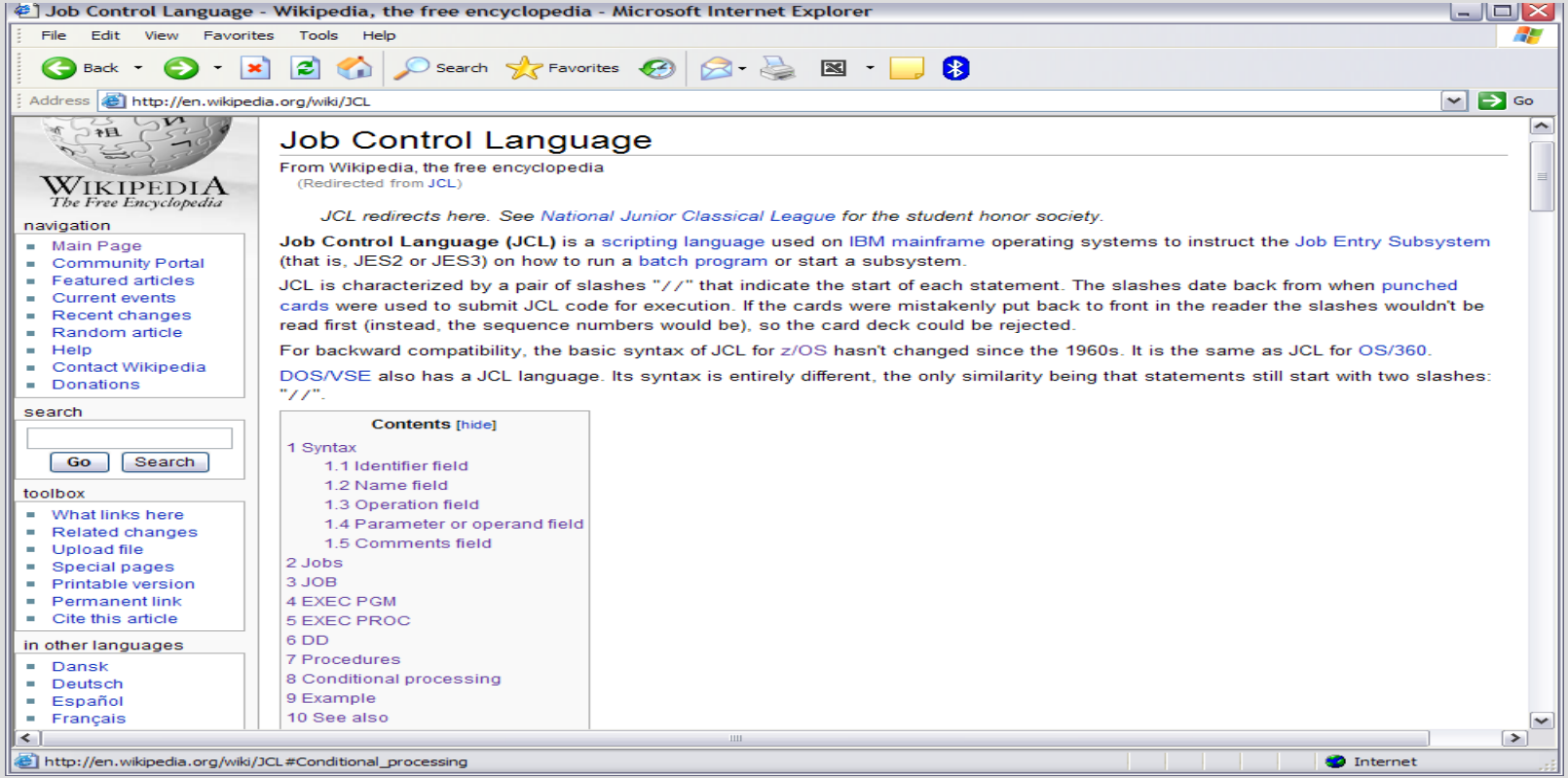
+ z/OS MVS

z/OS MVS JCL Reference	SA23-1385-40	30 Oct 2019	PDF **ieab600_v2r4.pdf** (3.22MB)
z/OS MVS JCL User's Guide	SA23-1386-40	10 Jul 2019	PDF **ieab500_v2r4.pdf** (1.43MB)

+ z/OS DFSMS

z/OS DFSMSdfp Utilities	SC23-6864-40	27 Apr 2020	PDF **idau100_v2r4.pdf** (3.41MB)

wikipedia.org



Unit summary

Having completed this unit, you should be able to:

- Understand purpose of JCL
- Understand JCL JOB, EXEC, and DD statements
- Understand relationship of program file name to JCL DDNAME
- Locate JCL professional manuals, documentation, and online help