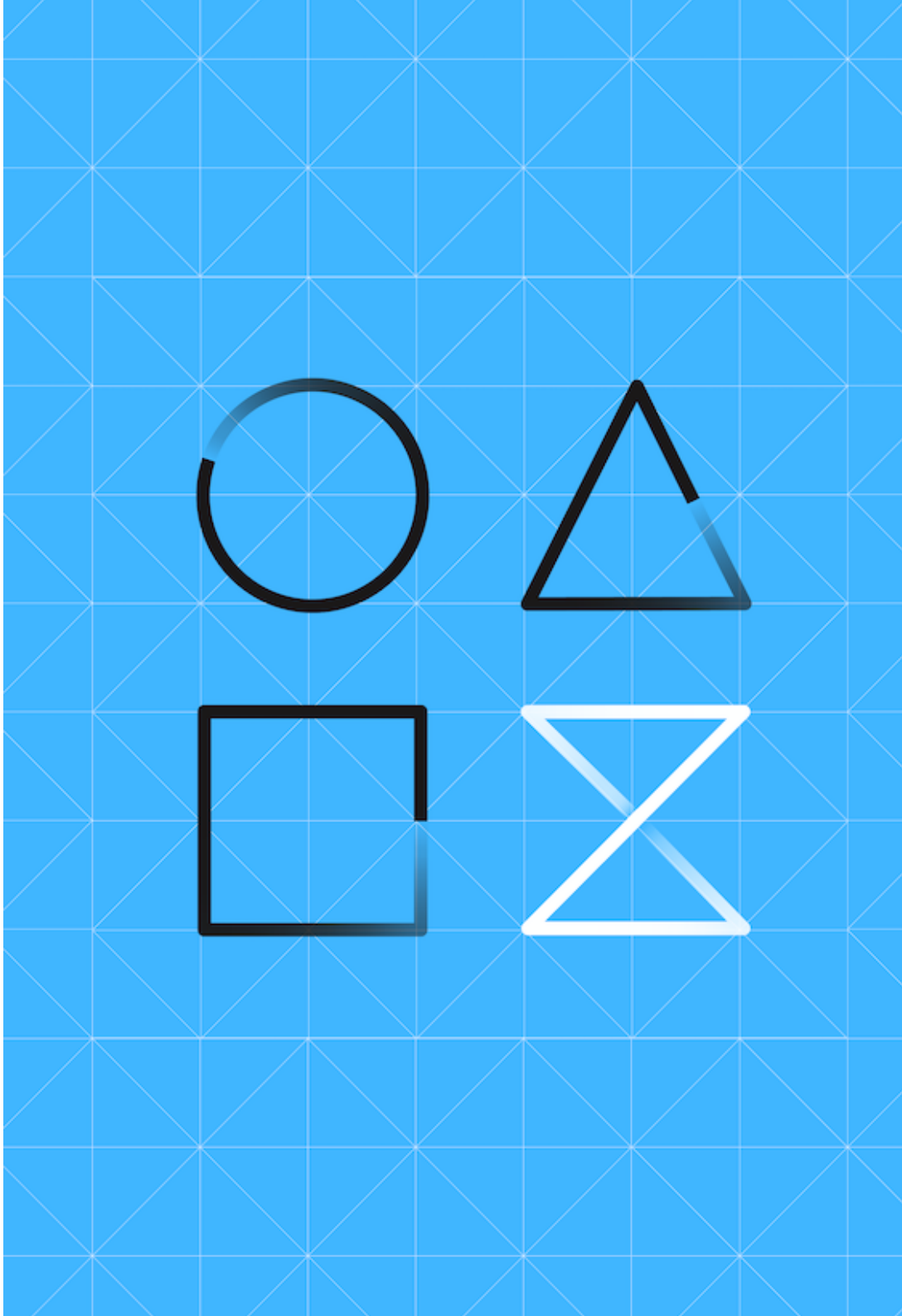
Fundamentals

230519-0916

JCL1

Orchestrating the Enterprise

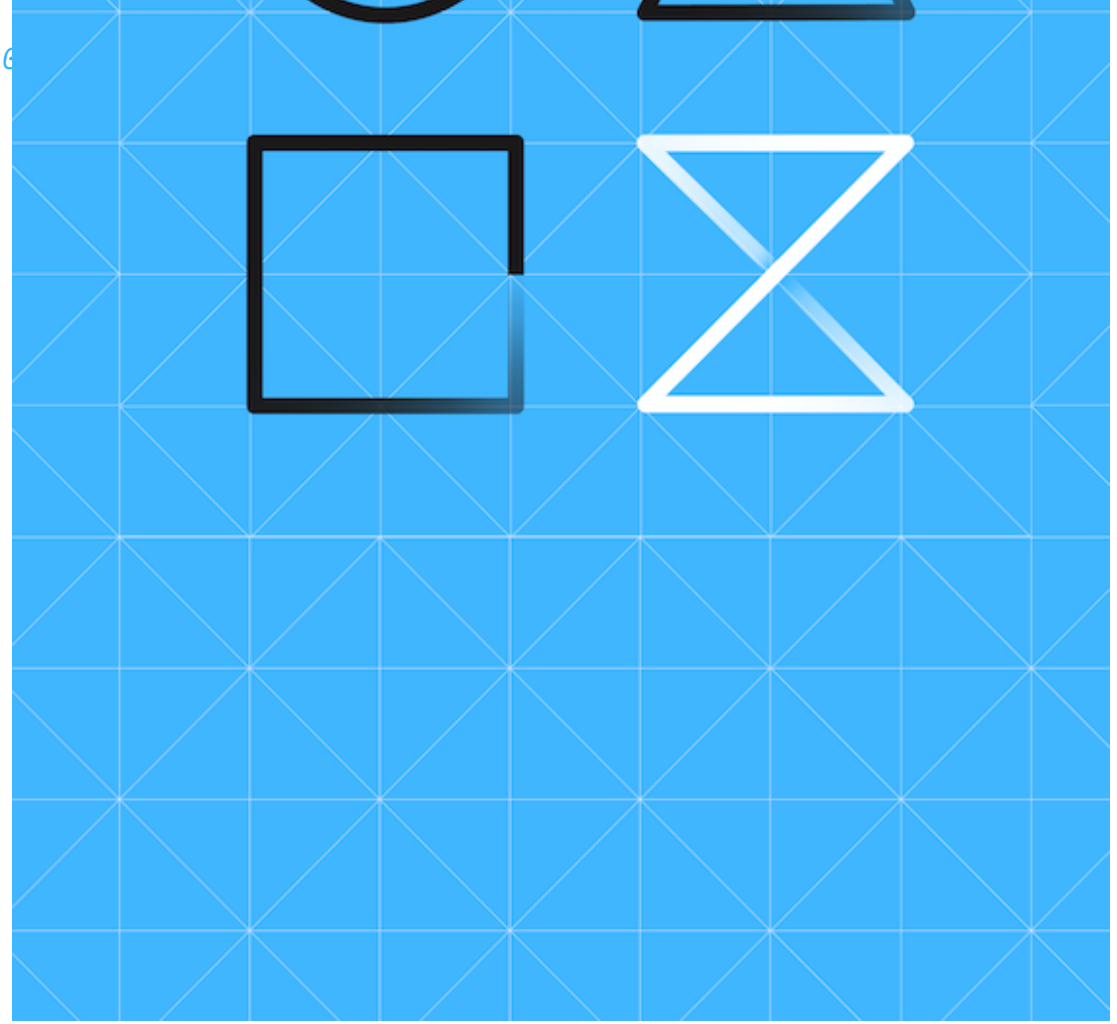
Automated Translation by Watson Language Translator



Z Xplore

JCL1

Orquestación de la empresa



JCL1-HACER QUE LAS COSAS SUCEDAN

El Reto

Ya ha visto algún JCL, pero realmente no hemos profundizado. En estos retos, aprenderemos un poco más sobre para qué se utiliza JCL, por qué es importante en un entorno Z y cómo puede llevar esas habilidades aún más lejos. JCL es una parte esencial de hacer que las cosas sucedan en z/OS y sentirse cómodo con los conceptos y la sintaxis le permitirá superar muchos retos a los que podría enfrentarse a medida que explora.

Antes De Empezar

Debería haber completado el reto FILES1, todo sobre conjuntos de datos y miembros. Si tienes esos conceptos entendidos, todos estarás listo para continuar con los pasos en este desafío.

Inversiones

Pasos	Duración
15	90 minutos

1 CARGAR INFORMACIÓN

ZXP.PUBLIC.JCL(JCLSETUP).jcl

```
1   //JCLSETUP JOB
2   //     EXEC PGM=IEFBR14
3   //LOAD   DD DSN=&SYSUID..LOAD,DISP=(,CATLG),DATACLAS=SLOAD
4   //JCL    DD DSN=&SYSUID..JCL,DISP=(,CATLG),DATACLAS=SPDS
5   //SOURCE DD DSN=&SYSUID..SOURCE,DISP=(,CATLG),DATACLAS=SPDS
6   //OUTPUT DD DSN=&SYSUID..OUTPUT,DISP=(,CATLG),DATACLAS=SPDS
7
```


Buscar en **ZXP.PUBLIC.JCL** para un miembro denominado **JCLSETUP** . Este es un trabajo bastante simple que asignará algunos conjuntos de datos nuevos que necesita para este y otros retos.

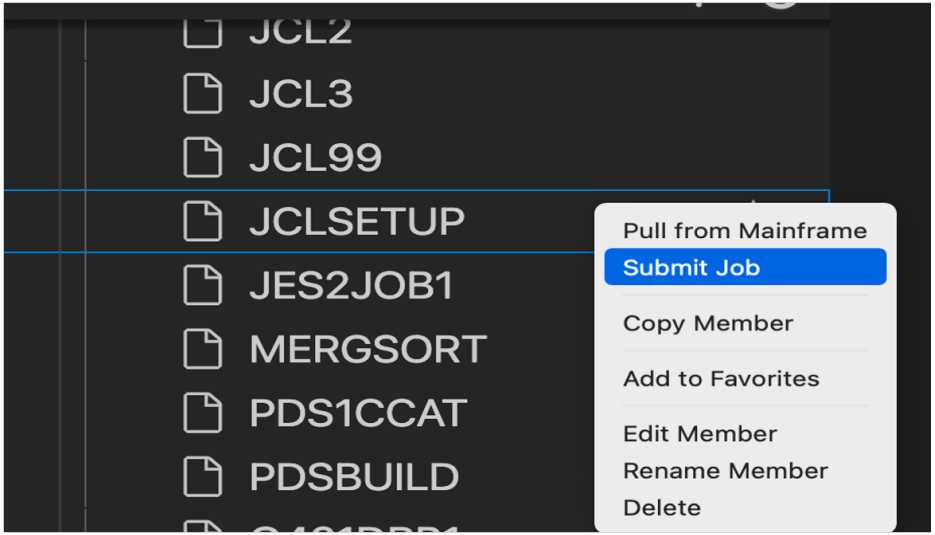
(La línea #4 crea su propio conjunto de datos JCL.)

Puede ver en la línea #3, menciona &SYSUID .. CARGA. El Ampersand (&) con SYSUID después de que es lo que se conoce como un "simbólico", y cuando el sistema ve que, *it* sustituirá &SYSUID por su ID de usuario.

Usted no necesita hacer ninguna oportunidad a este trabajo para que funcione.

Esto significa que todo el mundo puede utilizar el mismo trabajo y sustituirá automáticamente &SYSUID por su ID de usuario. ¡Qué conveniente!

2 SUBMIT IT



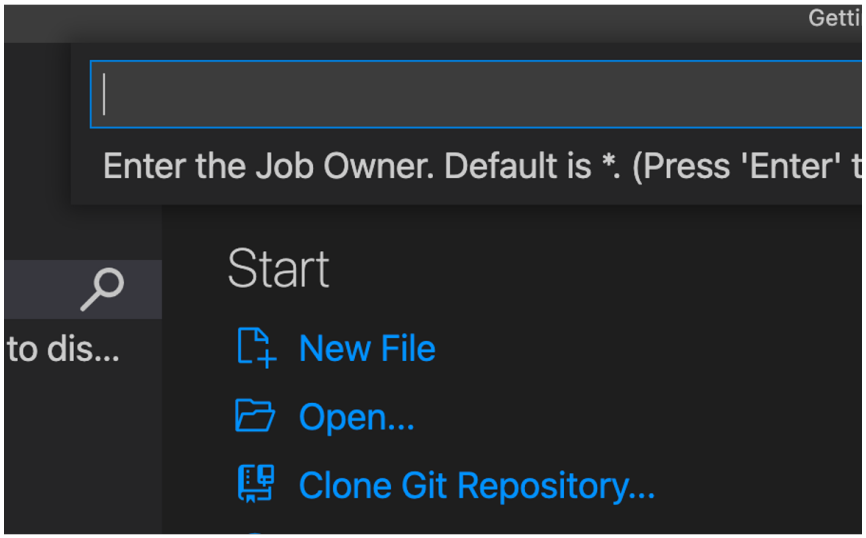
Pulse con el botón derecho del ratón en ese trabajo y seleccione Submit Job.

Una nota sobre la palabra "Trabajo": JCL se utiliza para describir al sistema exactamente lo que desea que haga. La tarea que entregamos al sistema se conoce como "Trabajo", y el componente de z/OS que acepta, procesa y gestiona la salida de estos trabajos se conoce como **Subsistema de entrada de trabajos (JES)** .

Por lo tanto, para este reto, hemos enviado un trabajo a JES para que procese la tarea que acabamos de mirar.

Nota : este trabajo está pensado para crear conjuntos de datos para usted; si lo ejecuta más de una vez, es probable que vea errores sobre **DUPLICAR** nombres de conjunto de datos.

3 FILTRAR Y BUSCAR



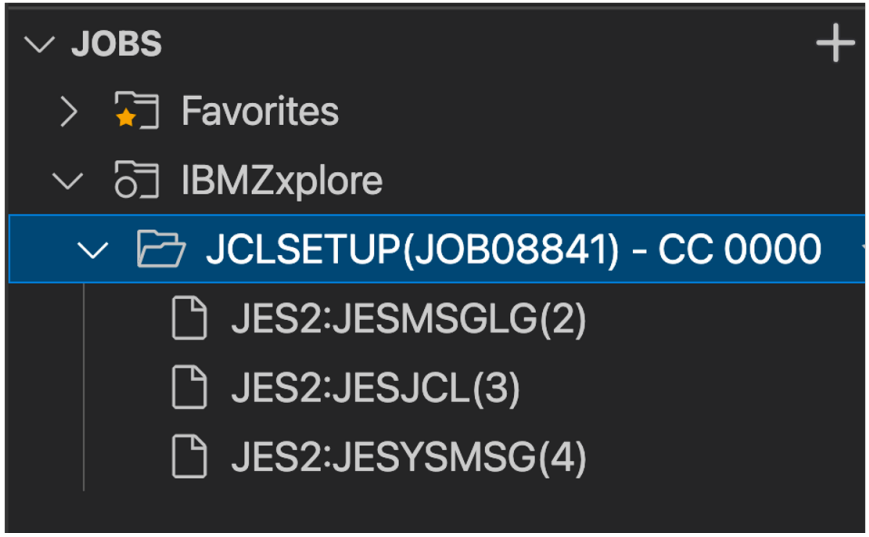
Ya debe tener un perfil en **TRABAJOS** en el lado izquierdo de VSCode.

Haga clic en la lupa ( ) a la derecha de ella.

- Especifique su ID de usuario para el propietario del trabajo
- Especifique un asterisco (*) para el prefijo de trabajo
- Pulse Intro (en blanco, sin datos) para la búsqueda de ID de trabajo.

Puede volver a hacer este filtro pulsando de nuevo en la lupa y seleccionando Propietario/Prefijo o ID de trabajo. Usted debe ser capaz de encontrar el trabajo que acaba de enviar aquí. Busque "JCLSETUP". Vamos a indagar en eso a continuación.

4 TIENES UN CERO. ¡PERFECTO!

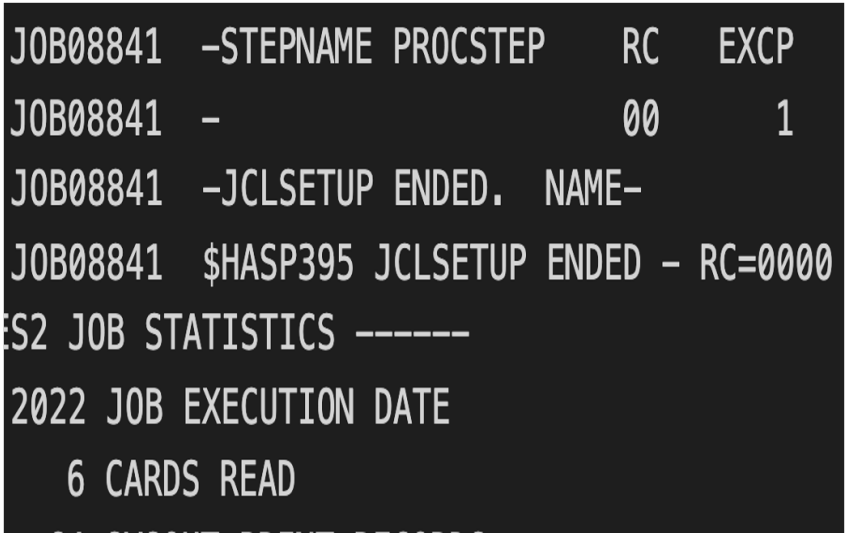


Abra el triángulo "twistie" junto al trabajo JCLSETUP que acaba de enviar. Probablemente habrá otros trabajos allí también, pero estamos buscando específicamente **JCLSETUP** . Si usted lo presentó más de una vez, encontrar el que tiene CC 0000 a la derecha.

También verá este número en el **JESMSGLG** miembro una vez que abra el twistie. Un código de condición (CC) de cero significa que todo se ejecutó como se esperaba, sin errores, ¡así que eso es bueno!

Si tienes cualquier otro número, entonces por lo general hay algo que vale la pena investigar y probablemente arreglar.

5 SALTAR A LA DERECHA



Es posible que haya observado que después de enviar JCL, aparece un pequeño mensaje en la esquina inferior derecha de VS Code. En lugar de cavar a través de la salida de JOBS, normalmente puede simplemente hacer clic en ese mensaje, y le llevará directamente a la salida.

Un trabajo empezará en **ACTIVO** mientras se ejecuta. Puede renovar el estado de un trabajo cerrando y, a continuación, volviendo a abrir el triángulo a la izquierda del nombre del trabajo.

6 INICIANDO ALGUNOS COBOL

```
1  //JCL2    JOB 1
2  //*************************************************/
3  //COBRUN  EXEC IGYWCL
4  //COBOL.SYSIN  DD DSN=ZXP.PUBLIC.SOURCE(CBL0001),DISP=SHR
5  //LKED.SYSLMOD DD DSN=&SYSUID..LOAD(CBL0001),DISP=SHR
6  //*************************************************/
7  // IF RC = 0 THEN
8  //*************************************************/
9  //RUN     EXEC PGM=CBL0001
10 //STEPLIB   DD DSN=&SYSUID..LOAD,DISP=SHR
11 //FNAMES    DD DSN=ZXP.PUBLIC.INPUT(FNAMES),DISP=SHR
12 //LNAMES    DD DSN=ZXP.PUBLIC.INPUT(LNAMES),DISP=SHR
13 //COMBINE   DD DSN=&SYSUID..OUTPUT(NAMES),DISP=SHR
14 //SYSOUT    DD SYSOUT=*,OUTLIM=15000
15 //CEEDUMP   DD DUMMY
16 //SYSUDUMP  DD DUMMY
```


Copie el **JCL2** miembro de **ZXP.PUBLIC.JCL** a los suyos **JCL** y, a continuación, abra la copia.

Es posible que tenga que cerrar y volver a abrir su **CONJUNTOS DE DATOS** triángulo para renovar la vista, para que se muestre JCL2.

Este JCL se utiliza para compilar y ejecutar algún código COBOL. Después de compilar, colocará el programa resultante en su **CARGA** conjunto de datos.

Nota : los programas del conjunto de datos LOAD son binarios-no podrá ver nada aquí con VSCode.

En JCL, las sentencias que definen de dónde provienen los datos, o a dónde van, se conocen como **Definición de datos** o, simplemente, "sentencias DD".

Busque la línea que empieza por **// COBRUN** -es el inicio de un "paso" de trabajo que con ejecutar el compilador COBOL. En la siguiente línea, puede ver el conjunto de datos de entrada (el origen) en la línea 18 (**// COBOL.SYSIN**), y dónde pondrá la salida en la siguiente línea (**// LKED.SYSLMOD**).

Las líneas que siguen al inicio del **// RUN** paso tienen el mismo formato:

```
//"ddname"  DD DSN="dataset",DISP="access"
```

- "ddname" también se conoce como el nombre de archivo-el nombre utilizado por los programas para acceder a los datos de los conjuntos de datos
- "dataset" es la ubicación real de los datos-esto puede cambiar, pero el programa no necesita ser consciente
- "access" indica cómo el programa puede utilizar el conjunto de datos

Leyendo más, si el trabajo obtiene un código de retorno de 0 (porque todo ha ido sin ningún problema) del paso de compilación, ejecutará el **CBL0001** programa.

¿Todos teniendo sentido hasta ahora? Estamos utilizando JCL para compilar y, a continuación, ejecutar algún código.

¿POR QUÉ SON IMPORTANTES JES Y JCL? ¿POR QUÉ NO PUEDO EJECUTAR PROGRAMAS?

Cuando somete JCL, va al subsistema de entrada de trabajos (abreviado a JES).

JES examina el JCL que ha enviado y recopila todos los recursos necesarios para realizar la tarea. En un sistema muy cargado, puede ser necesario priorizar algunos trabajos más bajos que otros para que el trabajo importante se haga más rápido.

Piense en JCL como el orden que escribe un camarero, y en JES como el personal de cocina que mira el orden y decide cómo lo van a manejar.

La L en JCL significa Lenguaje, pero realmente no es un lenguaje de programación tanto como es una forma para nosotros de describir eficazmente las tareas al sistema.

Todo lo demás que aparece en la salida del trabajo (el "registro de trabajo") es información sobre cómo se ejecutó el trabajo. Como puedes ver, hay un TON de información en aquí.

7 RUN, CODE, RUN

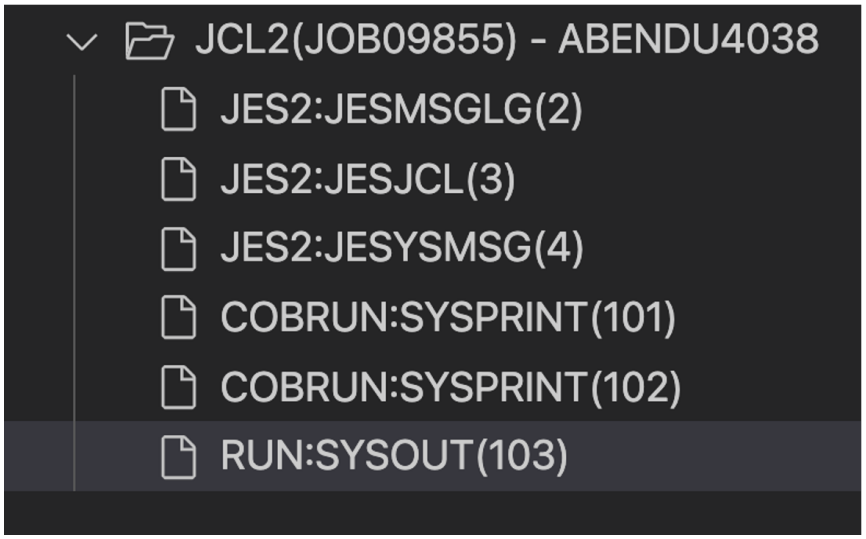
Después de compilar correctamente el código COBOL, JES ejecutará el programa (el **// RUN EXEC PGM=CBL0001** y dígale dónde se encuentran los conjuntos de datos de entrada, así como dónde se almacenará la salida en el conjunto de datos OUTPUT-

```
//COMBINE   DD DSN=&SYSUID..OUTPUT(NAMES),DISP=SHR
```

El nombre de la sentencia DD es lo que viene directamente después de las barras inclinadas dobles, por lo que "FNAMES" y "LNAMES", por ejemplo.

Cualquier línea que empiece por **//** * son comentarios y JES los ignora. Las líneas comentadas son útiles para proporcionar información informativa, o mantener líneas de código que podríamos utilizar más adelante, pero no es necesario en este momento.

8 NO PUEDEN SER TODOS CEROS



Enviar **JCL2** del conjunto de datos JCL y, a continuación, consulte la salida, utilizando lo que ha aprendido de los pasos anteriores de este reto.

Usted *se* obtener un **ABEND** (abreviatura de Abnormal End), así que algo no está del todo bien todavía.

Pero no se preocupe-con sus nuevas habilidades, usted llegará al fondo de esto!

En el paso siguiente, observará el código COBOL y verá cómo el código real coincide con el código JCL que se utiliza para compilarlo y ejecutarlo.

Y de nuevo, ¡no se preocupe! No es necesario que se convierta en un experto en COBOL para resolverlo; recuerde que es un **JCL** desafío, no un desafío COBOL.

9 COMPARAR EL CÓDIGO

```
*-----------------------
 IDENTIFICATION DIVISION.
*-----------------------
 PROGRAM-ID.    NAMES
 AUTHOR.        Otto B. Named
*--------------------
 ENVIRONMENT DIVISION.
*--------------------
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT FIRST-NAME ASSIGN TO FNAMES.
     SELECT LAST-NAME  ASSIGN TO LNAMES.
     SELECT FIRST-LAST ASSIGN TO COMBINED.
```


La sentencia JCL que empieza **// COBOL.SYSIN** apunta al código fuente COBOL que queremos compilar, así que empecemos por ahí. Abra ese código y empiece a ver el **CONTROL DE ARCHIVOS** área.

Aquí es donde obtenemos los nombres para el programa a los que necesitamos hacer referencia en nuestro JCL. Por ejemplo, **NOMBRE-PRIMERO** es a lo que hacemos referencia en el código COBOL, y que está asignado (o enlazado) al **NOMBRES** Sentencia DD en el JCL.

Abra el JCL, el código COBOL y el registro de trabajo; consulte la salida y podrá averiguar qué

*muy simple **soltero** cambio*

se debe realizar para que todo se enlace entre el COBOL y el JCL.

En realidad puede ayudar a dibujar todo en un pedazo de papel.

Cuando haya corregido el problema en JCL2, debería ejecutarse con un CC= 0, y *buscar la salida correcta en el miembro correcto* de su conjunto de datos OUTPUT.

Esta etapa podría tomar un poco de paciencia.

¿QUÉ SIGNIFICA COMPILAR? ¿QUÉ ES COBOL?

COBOL es un lenguaje de programación utilizado en muchas instituciones financieras, sanitarias y gubernamentales. Su alto grado de precisión matemática y los métodos de codificación directa hacen que sea un ajuste natural cuando los programas necesitan ser rápidos, precisos y fáciles de entender.

El código escrito por los humanos debe convertirse en código de máquina para que se ejecute como un programa. La compilación es un paso que realiza esta transformación. Nuestro JCL tiene dos pasos principales, compilar el código fuente en código de máquina y, a continuación, ejecutar el programa.

Este programa en concreto también requiere dos archivos de entrada y escribe en un archivo de salida, por lo que también especificaremos estos archivos (conjuntos de datos) en el JCL.

10 TODOS A BORDO

```
//*
//PEEKSKL EXEC PGM=IEBGENER
//SYSPRINT DD DUMMY
//SYSIN    DD DUMMY
//SYSUT1   DD *
Peekskill - 41mi
//SYSUT2   DD DSN=&SYSUID..JCL3OUT,DISP=(MOD,PASS,DELETE),
//            SPACE=(TRK,(1,1)),UNIT=SYSDA,
//            DCB=(DSORG=PS,RECFM=FB,LRECL=80)
//*
//CORTLNDT EXEC PGM=IEBGENER
//SYSPRINT DD DUMMY
//SYSIN    DD DUMMY
//SYSUT1   DD *
Cortlandt - 38mi
```


Copiar **JCL3** desde **ZXP.PUBLIC.JCL** en su propio conjunto de datos JCL. Cárguelo y eche un vistazo a lo que hay dentro. Verá que este JCL contiene una serie de pasos, cada uno de los cuales utiliza el **IEBGENER** programa de utilidad para dirigir un registro a un conjunto de datos secuencial.

Hay una cabecera, alguna información de la estación para las paradas de tren pico entre Poughkeepsie, NY y Grand Central Terminal en NYC, seguido por algún texto sobre las horas de operación.

Parece bastante sencillo ... vamos a ver cuál es la parte complicada.

11 UOU ' RE NO DUMMY

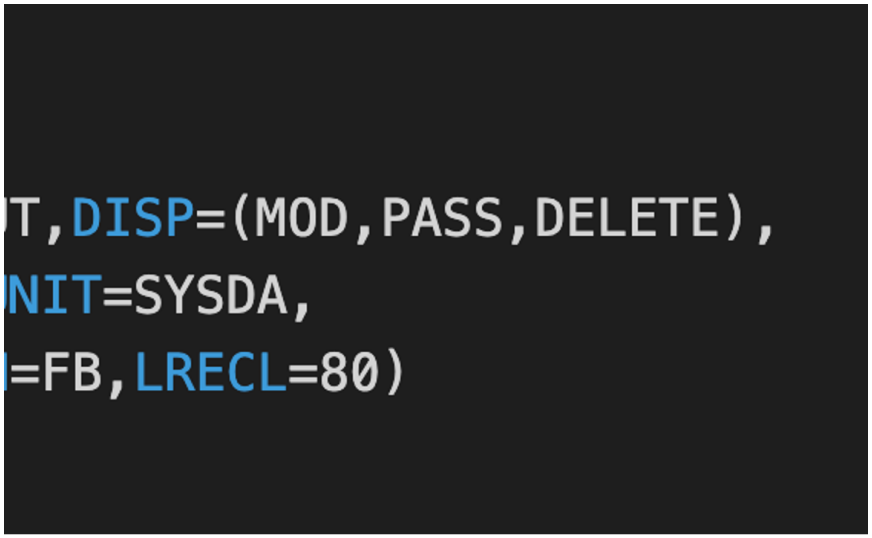
```
//JCL3     JOB
//*
//* IEBGENER is a system utility program to copy data
//*  where the default input  filename is SYSUT1
//*  and   the default output filename is SYSUT2
//*
//HEADER EXEC PGM=IEBGENER
//SYSPRINT DD DUMMY
//SYSIN    DD DUMMY
//SYSUT1   DD *
*****************************************
METRO NORTH POUGHKEEPSIE -> NYC M-F SCHEDULE
PEAK HOUR OPERATION
*****************************************
```


Es posible que también haya notado muchas menciones de **DUMMY** .

No se preocupe, este JCL no está llamando a nadie; es sólo una forma de decir "Este parámetro es necesario, pero esta vez no vamos a hacer nada con él, así que no importa".

Lo usamos aquí porque el programa que estamos ejecutando en cada paso (**IEBGENER**). requiere una sentencia de entrada y requiere una **SYSPRINT** Declaración DD, pero no vamos a hacer uso de ella, por lo que DUMMY es una forma de decir "No importa, no pierdas tu tiempo configurando esto"

12 UNA DISPOSICIÓN AMISTOSA



En cada fragmento de JCL que hemos utilizado hasta ahora, hemos encontrado algún tipo de **DISP** parámetro (disposición).

Los parámetros DISP se utilizan para describir cómo JCL debe utilizar o crear un conjunto de datos, y qué hacer con él después de que se complete el trabajo, o paso de trabajo.

Un parámetro DISP estándar tiene tres partes. El primer parámetro es el estado, que puede ser cualquiera de los siguientes:

Parámetro	Significado
NUEVO	Crear un nuevo conjunto de datos
SHR	Volver a utilizar un conjunto de datos existente y dejar que otras personas lo utilicen si lo desean
OLD	

Parámetro	Significado
	Volver a utilizar un conjunto de datos existente, pero no deje que otros lo utilicen mientras lo estamos utilizando
MOD	Sólo para conjuntos de datos secuenciales. Vuelva a utilizar un conjunto de datos existente, pero solo añada nuevos registros al final del mismo. Si no existe ningún conjunto de datos, cree uno nuevo.

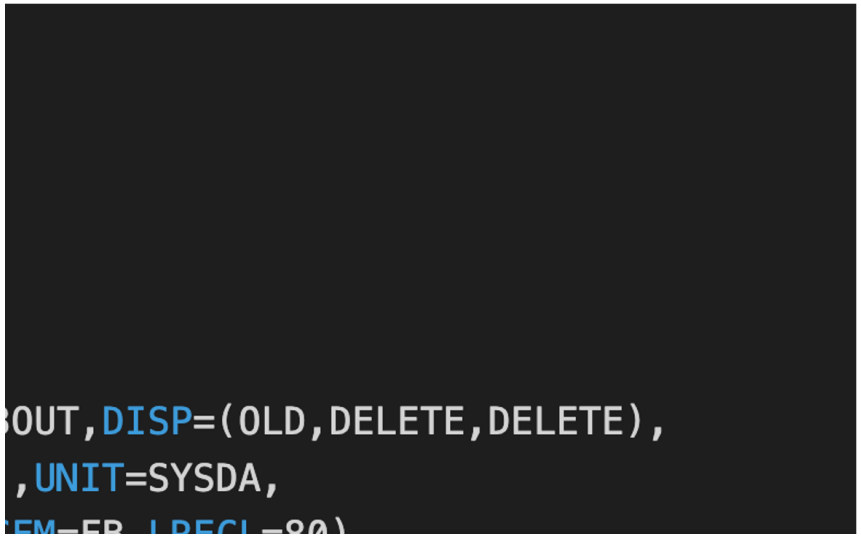
¿SALIDA DE TRABAJO? ¿SENTENCIAS DD? AYÚDAME A ENTENDER, POR FAVOR

Cuando se ejecuta un trabajo, genera una salida. Esto puede incluir los datos que está buscando, las razones por las que un trabajo no se ha ejecutado correctamente, o tal vez sólo información sobre el sistema y los pasos que ha realizado para que el trabajo se lleve a cabo. Aquí hay mucha información que probablemente *no lo haga* , pero siempre es mejor tenerlo que confundirse sobre el estado de un trabajo importante.

Una gran parte del JCL son las sentencias DD, que especifican qué conjuntos de datos (y miembros) se deben utilizar para la entrada y salida del trabajo que está sometiendo al sistema.

El DD en la sentencia DD representa la definición de datos y empiezan con // ddname. Especificarán el nombre del conjunto de datos (o miembro), si ya existe o debe crearse (conocido como la disposición), dónde debe ir la salida, cuánto espacio debe ocupar y otras variables.

13 ¿CUÁL ES SU ESTADO?



El campo 2 del parámetro DISP describe qué debe pasar con el conjunto de datos en el caso de una finalización normal, y el tercer campo es qué debe pasar con él en el caso de una anomalía.

Hay una serie de valores que podemos utilizar aquí, pero para este reto, sólo necesita saber lo siguiente:

Campo2	Significado
SUPRIMIR	Borrarlo del almacenamiento por completo
CATLG	Registre el conjunto de datos para que podamos utilizarlo más adelante
PASS	Una vez completado este paso, mantenga pulsado para que los pasos que vienen después de éste puedan utilizarlo

14 JUSTO A TIEMPO

```
1  ****************************************
2  METRO NORTH POUGHKEEPSIE -> NYC M-F SCHEDULE
3  PEAK HOUR OPERATION
4  ****************************************
5  Poughkeepsie - 74mi
6  New Hamburg - 65mi
7  Beacon - 59mi
8  Cold Spring - 52mi
9  Garrison - 50mi
10 Peekskill - 41mi
11 Cortlandt - 38mi
12 Croton-Harmon - 33mi
13 Harlem - 125th Street - 4mi
14 Grand Central Terminal - 0mi
15 ************************************************************
16 Peak fares are charged during business rush hours on any
17 weekday train scheduled to arrive in NYC terminals between
18 6 a.m. and 10 a.m. or depart NYC terminals between 4 p.m.
19 and 8 p.m. On Metro-North trains, peak fares also apply to
20 travel on any weekday train that leaves Grand Central Terminal
21 between 6 a.m. and 9 a.m.
22 Off-peak fares are charged all other times on weekdays, all
23 day Saturday and Sunday, and on holidays.
```

Envíe su copia del **JCL3** y mire la salida.

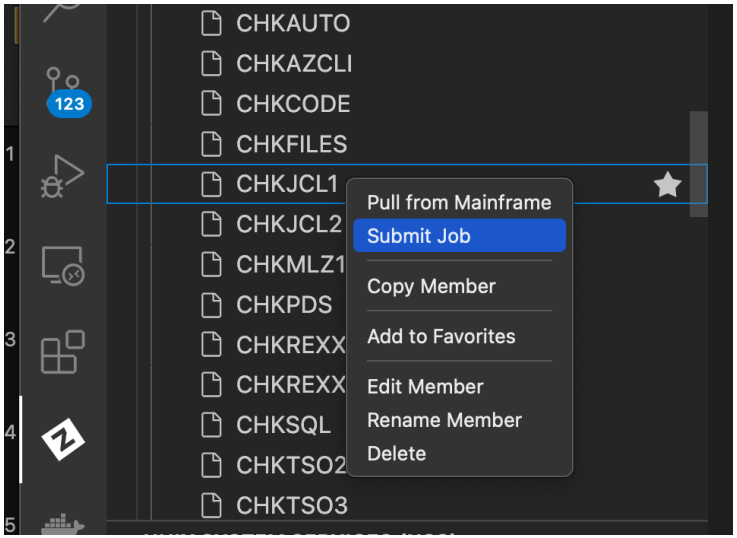
Hay dos ediciones que debe realizar para que este trabajo se ejecute al 100% correctamente:

- una lista completa de **10 paradas** , de Poughkeepsie a Grand Central Terminal
- la información en la parte superior e inferior del conjunto de datos.

También debe tener ***sin paradas de repetición*** . Si Beacon o Cortlandt se enumeran en dos veces, algo todavía necesita arreglo.

Cuando se complete, deberá tener la salida completa en su **JCL3OUT** conjunto de datos secuencial, totalizando 23 líneas (registros).

15 ¿QUIÉN NUEVO?



Ahora envíe el trabajo **CHKJCL1** desde **ZXP.PUBLIC.JCL** para validar la salida correcta de JCL2 y JCL3, y esperar ver el código de finalización (CC) de 0.

Si CHKJCL1 devuelve CC=0127, vuelva atrás y efectúe una doble comprobación y ajuste el trabajo para JCL2 y JCL3, y vuelva a someter estos trabajos si es necesario.

Vuelva a comprobar la salida correcta enviando **CHKJCL1** de nuevo hasta que obtenga CC=0000.

Tendrá que suprimir el **JCL3OUT** conjunto de datos de salida cada vez antes de volver a enviar **JCL3** .

Usted ha logrado mucho, y con suerte entender el requisito para seguir los detalles ... ¡Bien hecho!

Buen trabajo-recapitulemos	Siguiente ...
JCL puede parecer un poco complicado, y tal vez incluso un poco innecesario al principio. No estamos acostumbrados a usar código para iniciar programas, por lo general, sólo haga doble clic en ellos y se ejecutan! Sin embargo, una vez que usted comienza a entrar en los tipos de aplicaciones que mantienen los sistemas Z ocupados 24/7, usted comienza a construir una apreciación de la precisión y la potencia que la estructura le da. Basta con decir que JCL es una habilidad necesaria para cualquier verdadero profesional de Z.	Descubra el entorno Unix dentro de zOS-Unix System Services (USS)

JCL1|230519-0916