

AYI ACADEMY: TALEND

Material práctico



Módulo 6: Control de la ejecución

Módulo 6: Control de la ejecución

- **Temas:**
- Configurar el entorno
- Administrar archivos
- Procesamiento de archivos
- Administrar la ejecución de un Job usando un master Job
- Resumen

Objetivos

Después de completar este módulo, podrá:

- Usar componentes para enumerar, archivar y eliminar archivos de un directorio
- Administrar iteraciones dentro de un Job
- Reutilizar variables de un componente para configurar otros componentes
- Usar triggers para conectar componentes y subJobs
- Crear un master Job
- invalidar variables de contexto para subJobs
- Exportar un master Job y sus dependencias

Ejercicio

Este módulo, comienza con un Job diseñado para exportar datos de una base de datos a varios archivos delimitados almacenados en el mismo directorio. Lo que hará será diseñar un segundo Job para procesar todos estos archivos almacenados al mismo tiempo. Luego, administrará estos dos Jobs desde un master Job y utilizará diferentes opciones de configuración.

Cuando termine el master Job, lo exportará con todas sus dependencias para que pueda proporcionar el paquete completo a otra persona que pueda necesitar ejecutarlo.

Tema 1: Configurar el entorno

Antes de comenzar

Antes de desarrollar proyectos en Talend Studio, inicie Studio y abra un proyecto en su espacio de trabajo.

Opening a project

1.- Inicie Talend Studio utilizando el icono de Studio.

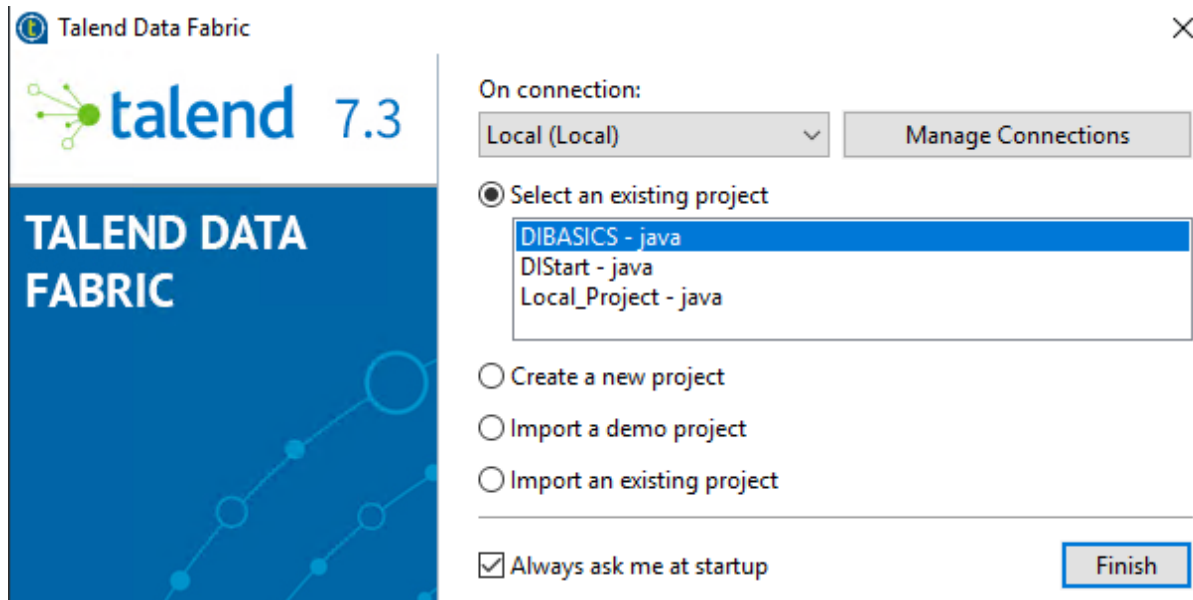


Se muestra Talend Data Fabric, que le permite abrir un proyecto existente o crear uno nuevo.

2.- Abra el proyecto DIBASICS.

Confirme que el proyecto **DIBASICS** esté disponible en la lista de proyectos existentes.

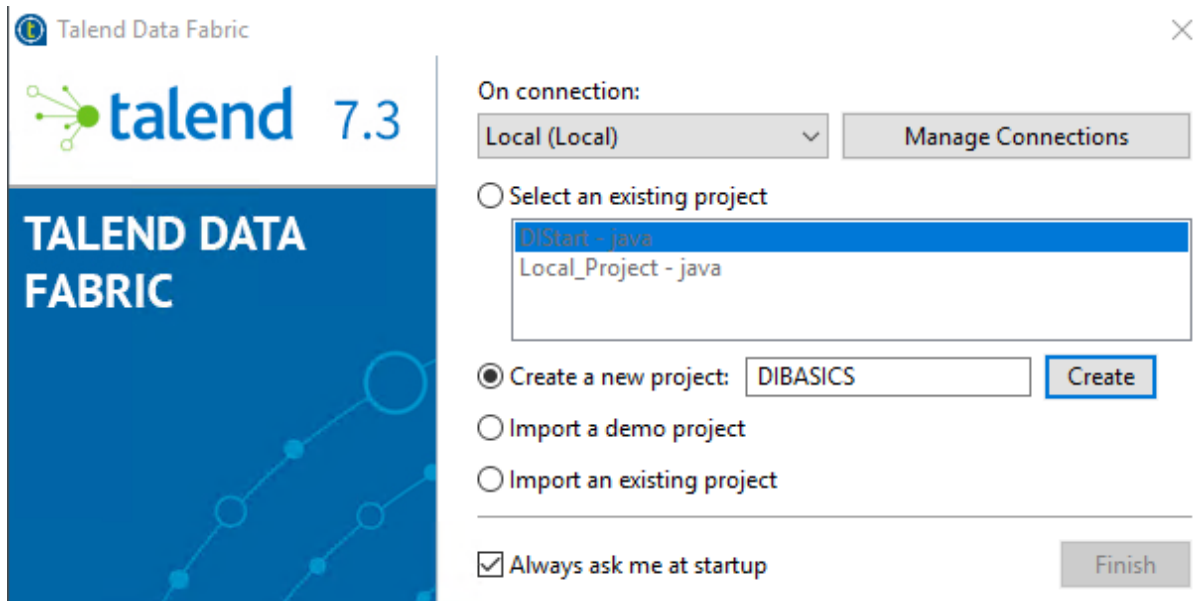
→ Si el proyecto **DIBASICS** aparece en la lista, selecciónelo y presione **Finish**.



→ Si el proyecto no existe, siga los siguientes pasos para crearlo:

a.- Para **On connection**, seleccione **Local (Local)**.

b.- Click en **Create a new project** e ingrese *DIBASICS*



c.- Click **Create** y espere a que el proyecto aparezca en la lista.

d.- Presione **Finish**.

Configurar el proyecto

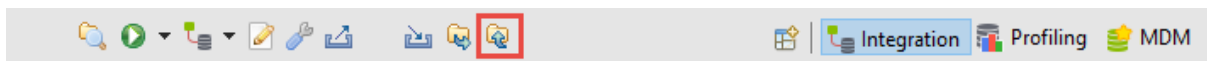
1.- En **Quick Access**, asegúrese de que la perspectiva **Integration** está seleccionada.



2.- Cargue el archivo.

El archivo ImportExecutionControl.zip se almacena en C:\StudentFiles\DIBasics\ExecutionControl. Cárguelo para importar un Job y sus metadatos relacionados al repositorio.

a.- En **Quick Access**, click en el ícono **folder and up arrow**.

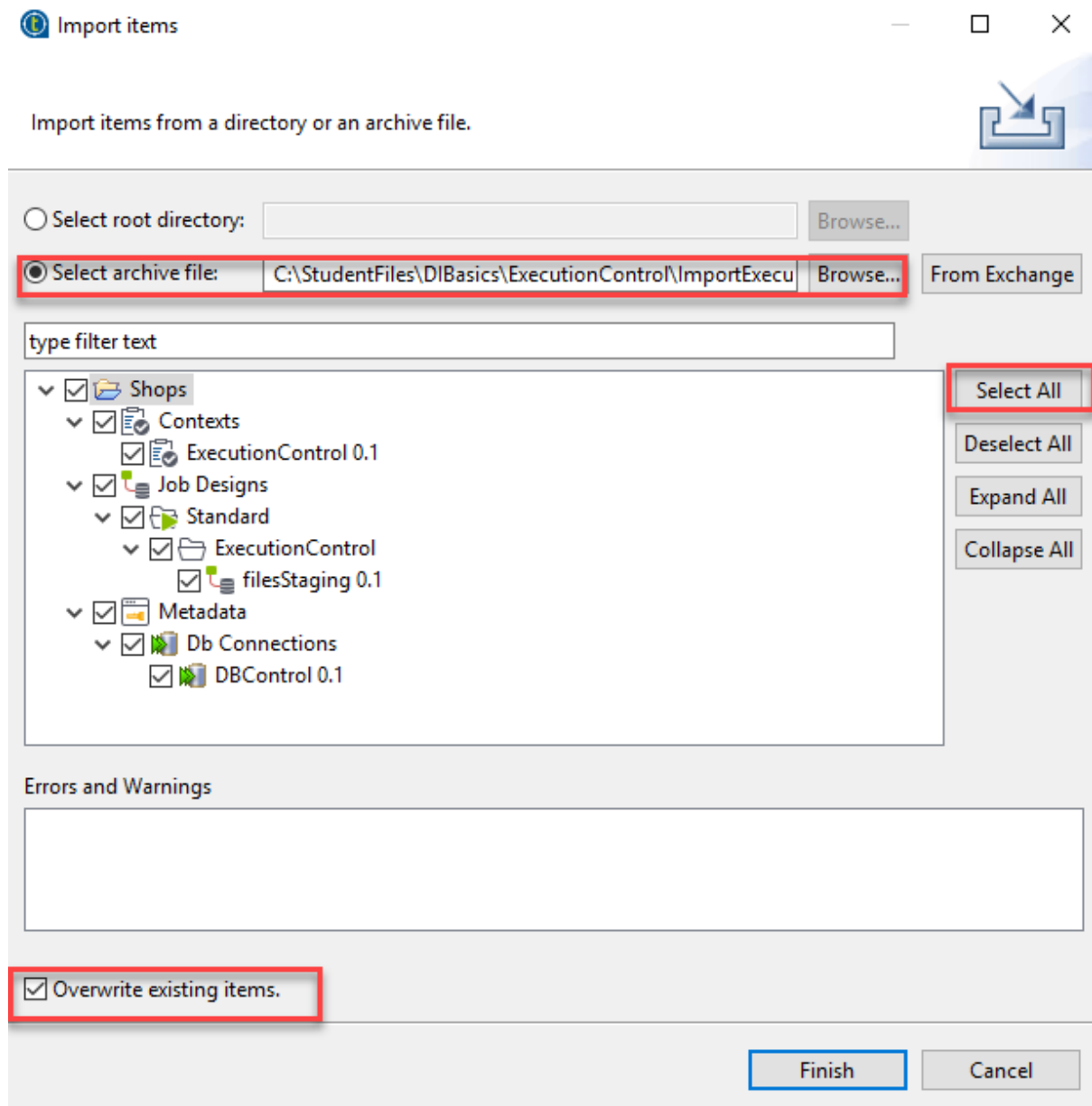


b.- En la ventana **Import items**, click en **Select archive file**, luego presione **Browse**.

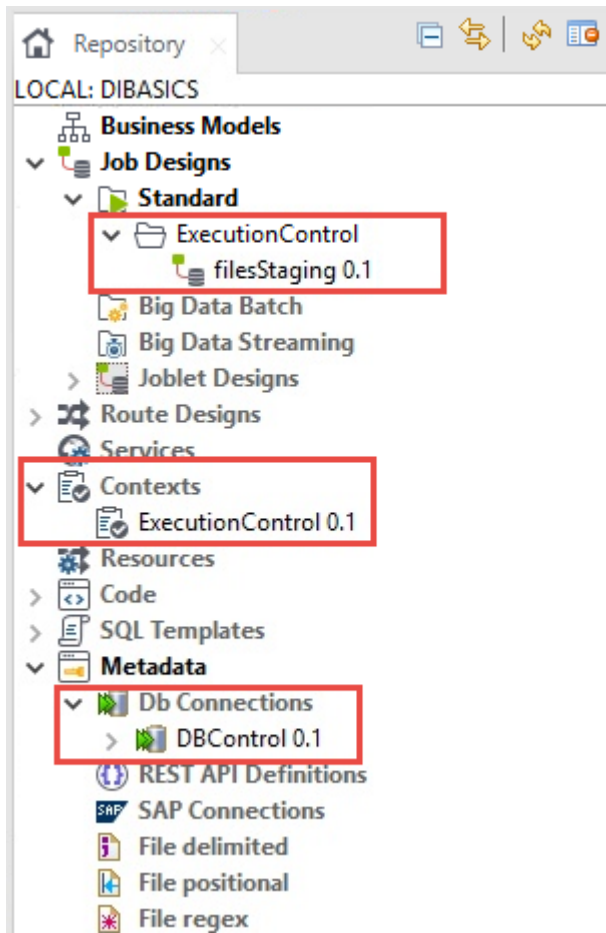
c.- Navegue hasta **C:\StudentFiles\DI Basics\ExecutionControl** y abra el archivo **ImportExecutionControl.zip**.

d.- Click en **Select All**.

e.- Asegúrese de marcar la opción **Overwrite existing items**, y presione **Finish**.



Ha importado un Job, un context group, y metadata para la conexión a la base de datos, al repositorio.



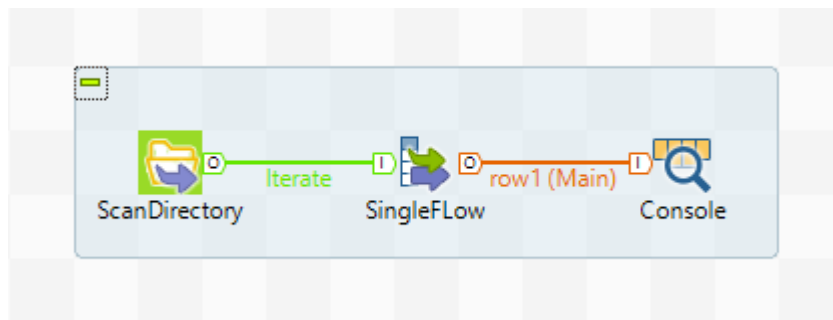
El Job se almacena en la carpeta ExecutionControl en Repository > Job Designs > Standard.

Tema 2: Administrar archivos

Antes de comenzar

Este ejercicio le enseña cómo realizar algunas operaciones básicas con archivos. Primero ejecuta el Job que importó para crear varios archivos en un directorio. Usando iteraciones y variables de componentes, escanea el contenido del directorio e imprime la lista de archivos en la consola

Al finalizar, su Job debería verse así:

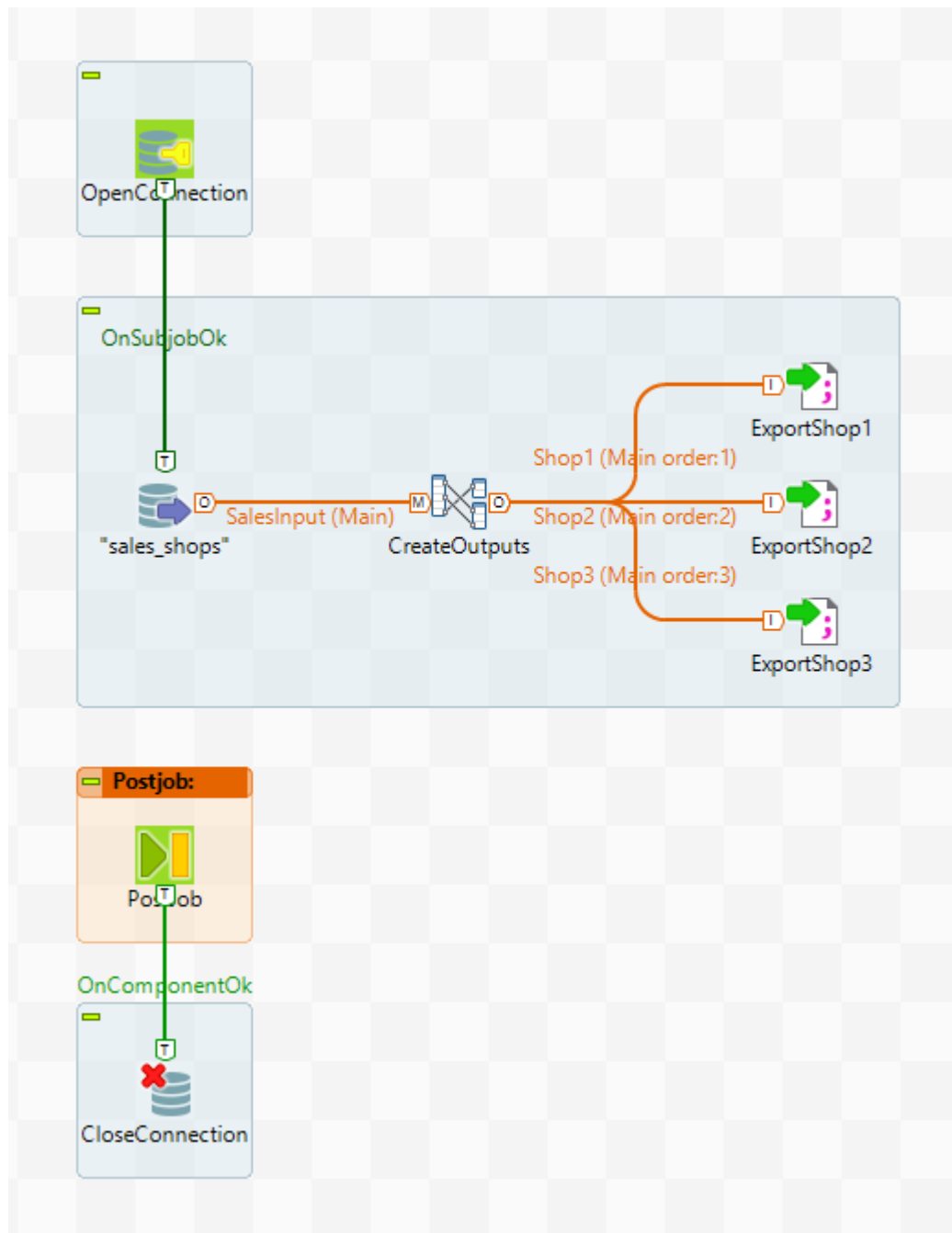


Crear archivos

Ejecute el Job filesStaging que importó para crear los archivos en un directorio.

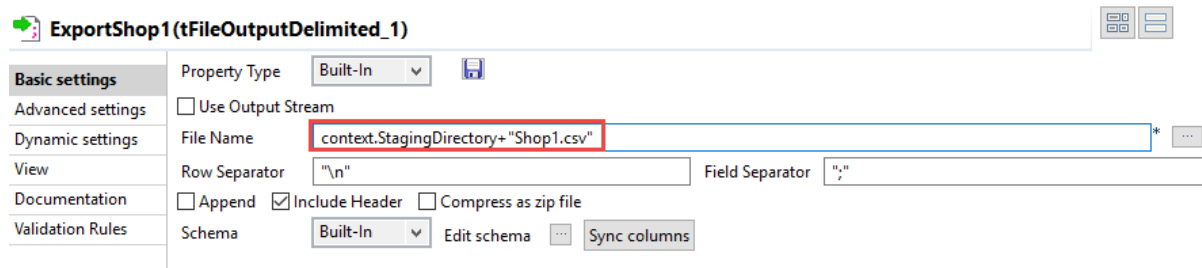
1.- En **Designer**, abra el Job y examine su estructura. Luego ejecútalo.

a.- En **Repository**, expanda **Job Design > Standard > ExecutionControl**, luego haga doble click en **filesStaging**.



El Job extrae datos desde una base de datos MySQL. Usando un componente tMap, crea tres salidas distintas, las cuales guardan localmente un archivo delimitado.

b.- Doble click en **ExportShop1**. En **File Name**, puede ver que los datos se exportan en un archivo CSV y se almacenan en un directorio definido por una variable de contexto.



c.- Abra la pestaña **Contexts**. El Job se puede ejecutar en dos contextos distintos: Development y Production. Las credenciales de la base de datos difieren según el contexto seleccionado.

La ruta de almacenamiento es: C:/StudentFiles/DIBasics/ExecutionControl/Staging.

	Name	Type	Comment	Production	Development	Prompt
				Value	Value	
1	ExecutionControl (from repository c					
2	DBControl_AdditionalParams	String		noDatetimeStringSync=true	noDatetimeStringSync=true	DBControl_Additic
3	DBControl_Database	String		production	training	DBControl_Databa
4	DBControl_Login	String		talend	talend	DBControl_Login
5	DBControl_Password	Password		*****	*****	DBControl_Passwc
6	DBControl_Port	String		3306	3306	DBControl_Port?
7	DBControl_Server	String		localhost	localhost	DBControl_Server
8	StagingDirectory	Directory		C:/StudentFiles/DIBasics/ExecutionControl/Staging/	C:/StudentFiles/DIBasics/ExecutionControl/Staging/	StagingDirectory

d.- Abra la pestaña **Run**. Seleccione el contexto **Development**.

Click **Run**. El Job debe finalizar con **[exit code=0]**.

Job filesStaging

- Basic Run
- Debug Run
- Advanced settings
- Target Exec
- Memory Run

Execution

Run Kill Clear

Starting job filesStaging at 14:24 12/03/2021.
[statistics] connecting to socket on port 3414
[statistics] connected
[statistics] disconnected
Job filesStaging ended at 14:24 12/03/2021. [Exit code = 0]

Development

Name	Value
DBControl_Additio...	noDatetimeString...
DBControl_Database	training
DBControl_Login	talend
DBControl_Password	****
DBControl_Port	3306
DBControl_Server	localhost
StagingDirectory	C:/StudentFiles/DI...

2.- Abra los archivos y confirme que todos tienen la misma estructura.

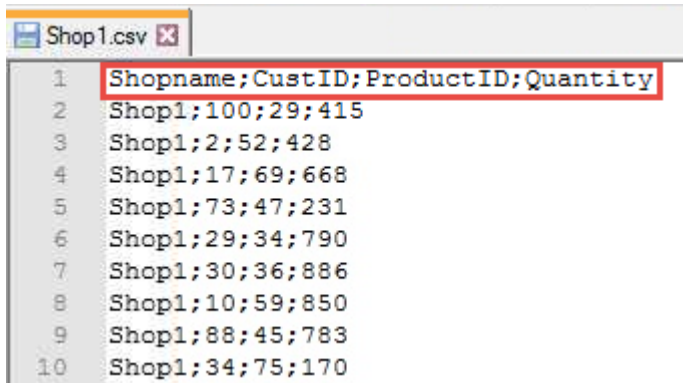
a.- Usando el explorador de archivos, navegue hasta C:/StudentFiles/DIBasics/ExecutionControl/Staging.

File Home Share View				
This PC > Local Disk (C:) > StudentFiles > DIBasics > ExecutionControl > Staging				
Name	Date modified	Type	Size	
Shop1.csv	11/22/2018 5:11 AM	CSV File	77 KB	
Shop2.csv	11/22/2018 5:11 AM	CSV File	77 KB	
Shop3.csv	11/22/2018 5:11 AM	CSV File	77 KB	

Se han creado tres archivos en el directorio.

b.- Haga click derecho en un archivo y seleccione **Edit with Notepad++**.

La fila de encabezado indica que los datos se dividen en cuatro columnas.



1	Shopname;CustID;ProductID;Quantity
2	Shop1;100;29;415
3	Shop1;2;52;428
4	Shop1;17;69;668
5	Shop1;73;47;231
6	Shop1;29;34;790
7	Shop1;30;36;886
8	Shop1;10;59;850
9	Shop1;88;45;783
10	Shop1;34;75;170

Todos los archivos comparten la misma estructura. También son del mismo tamaño: 5.000 filas de datos.

Escanear el contenido de un directorio

En este ejercicio, explorará el contenido del directorio Staging utilizando el componente tFileList, que genera la lista de archivos como iteraciones. Este componente no admite salidas de fila principal; solo se puede usar al comienzo de un flujo de datos y se puede conectar a un componente con una conexión de iteración. A diferencia de otras conexiones, la conexión Iterate no transfiere datos. En cambio, como el componente tFileList escanea los elementos en un directorio, le permite realizar cada iteración en un solo elemento (en este caso, para cada archivo descubierto en el directorio).

1.- Cree un nuevo standard Job en **Job Design > Standard > ExecutionControl** y nómbrelo *listDirectory*.

New Job
Add a job in the repository

Name: listDirectory

Purpose: List files stored in a directory

Description: Files are generated using the filesStaging Job

Author: training@talend.com

Locker:

Version: 0.1 M m

Status:

Path: ExecutionControl Select

Finish Cancel

2.- Importe el grupo de contexto **ExecutionControl** a su Job.

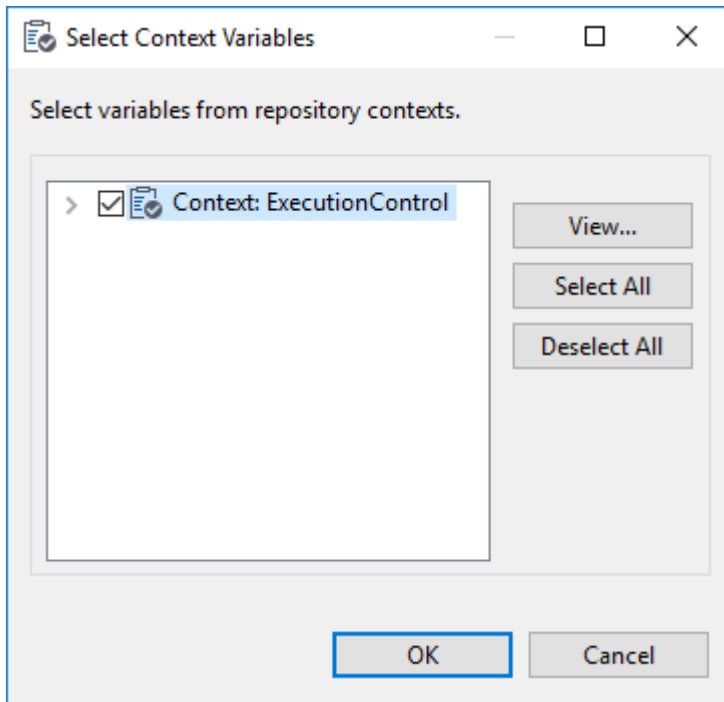
a.- Abra la pestaña **Contexts**.

b.- Debajo de la tabla, click en el botón **Context**:

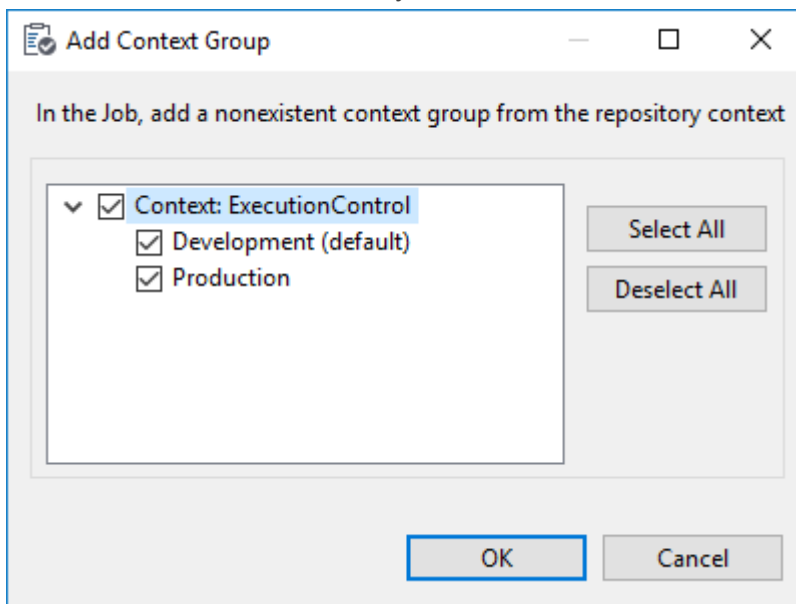
Name	Type	Comment	Default Value

Default context environment: Default

c.- Seleccione el contexto **ExecutionControl** y presione **OK**.



d.- Click en **Select All** y **OK**.

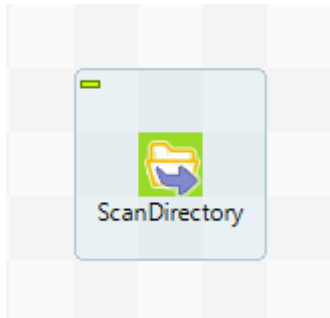


Los contextos Development y Production deben haber sido importados al Job.

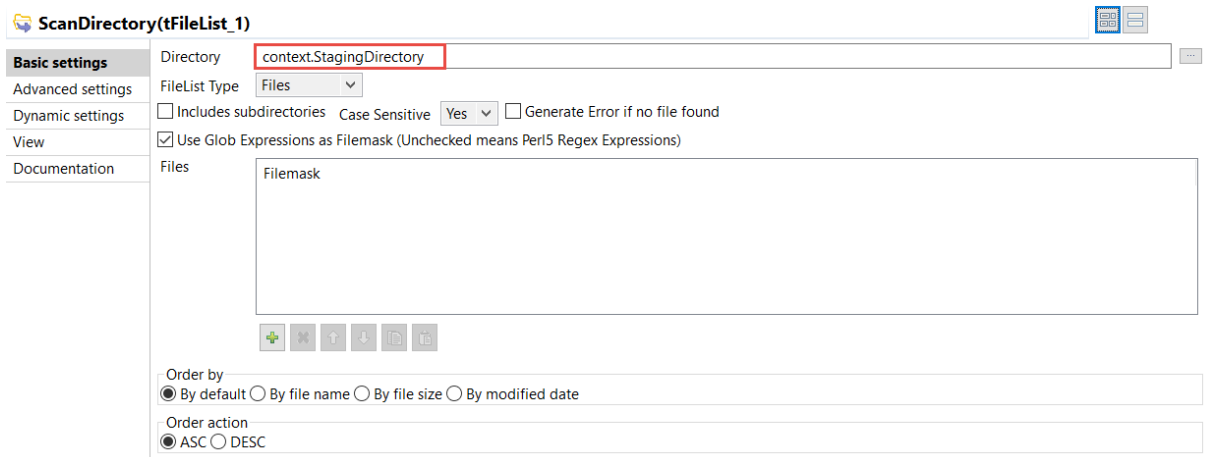
	Name	Type	Comment	Production	Development	
				Value	Value	Prompt
1	ExecutionControl (from repository c					
2	DBControl_AdditionalParams	String		noDateTimeStringSync=true	noDateTimeStringSync=true	DBControl_Additic
3	DBControl_Database	String		production	training	DBControl_Databa
4	DBControl_Login	String		talend	talend	DBControl_Login
5	DBControl_Password	Password		*****	*****	DBControl_Passwc
6	DBControl_Port	String		3306	3306	DBControl_Port?
7	DBControl_Server	String		localhost	localhost	DBControl_Server
8	StagingDirectory	Directory		C:/StudentFiles/DIBasics/ExecutionControl/Staging/	C:/StudentFiles/DIBasics/ExecutionControl/Staging/	StagingDirectory;

3.- Agregue un componente tFileList y configúrelo para escanear el directorio Staging.

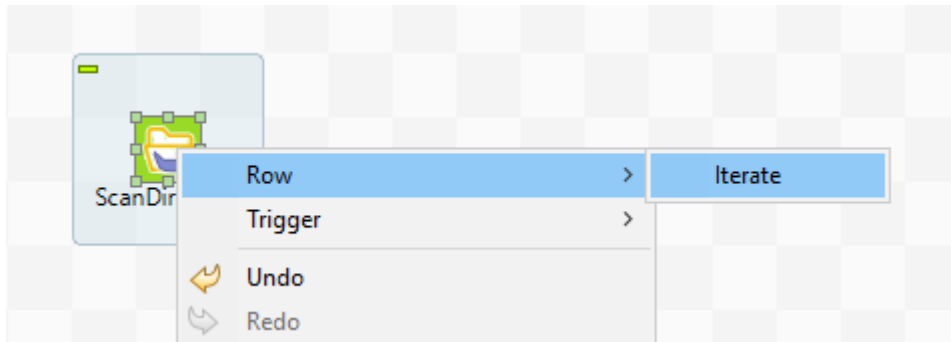
a.- En **Designer**, añada un componente **tFileList** y nómbrelo *ScanDirectory*.



b.- Haga doble click en el componente **ScanDirectory**. Para el parámetro **Directory**, ingrese *context.StagingDirectory*.

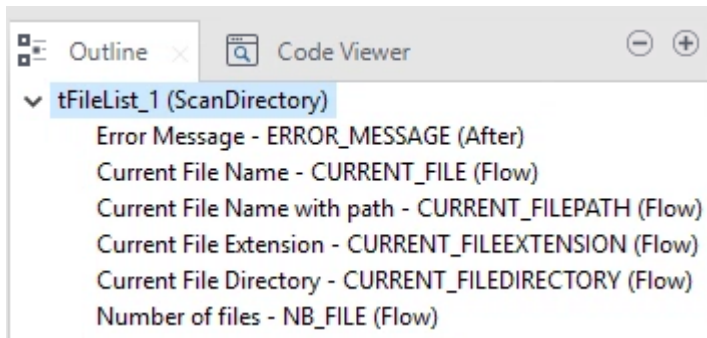


c.- En **Designer**, haga click derecho en **ScanDirectory** y seleccione **Row**.



Para este componente, solo está disponible una conexión de iteración.

4.- Para mostrar las variables asignadas en los componentes del Job, debajo de **Repository**, abra la pestaña **Outline**.



Estas variables se pueden utilizar para configurar otros componentes en el Job. Durante la ejecución del Job, los valores de las variables cambian. Por ejemplo, cuando el componente tFileList escanea los archivos en un directorio, la variable de la ruta del archivo actual toma un nuevo valor para cada iteración.

Usar variables del componente

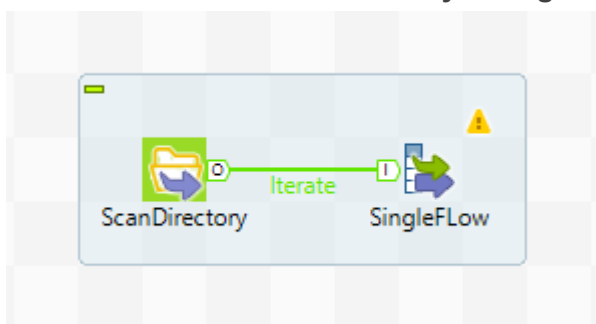
Tiene dos opciones: conectar un componente para procesar por separado las iteraciones generadas por el componente tFileList o convertir estas iteraciones en un solo flujo. Procesará las iteraciones en el próximo ejercicio.

1.- Conecte el componente.

Para convertir las iteraciones en un solo flujo, use un componente tIterateToFlow y configúrelo usando una variable de componente de Scandirectory.

a.- Añada un componente **tIterateToFlow** y nómbrelo *SingleFlow*.

b.- Conecte **ScanDirectory** a **SingleFlow** usando **Iterate** row.



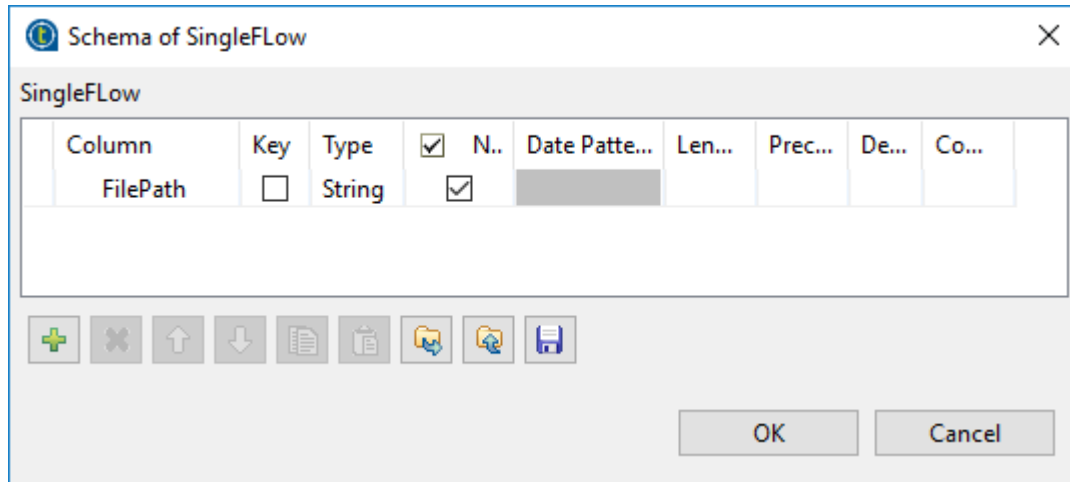
💡 A diferencia de otros tipos de conexiones, el nombre de la conexión Iterate es de solo lectura.

2.- El esquema no se puede heredar del componente anterior, por lo que debe crear una sola columna String.

a.- Doble click en **SingleFlow**.

b.- A la derecha de **Edit schema**, click [...].

c.- Click [+] para agregar una nueva columna al schema. Nombre la columna *FilePath*. Para **Type**, seleccione **String**.



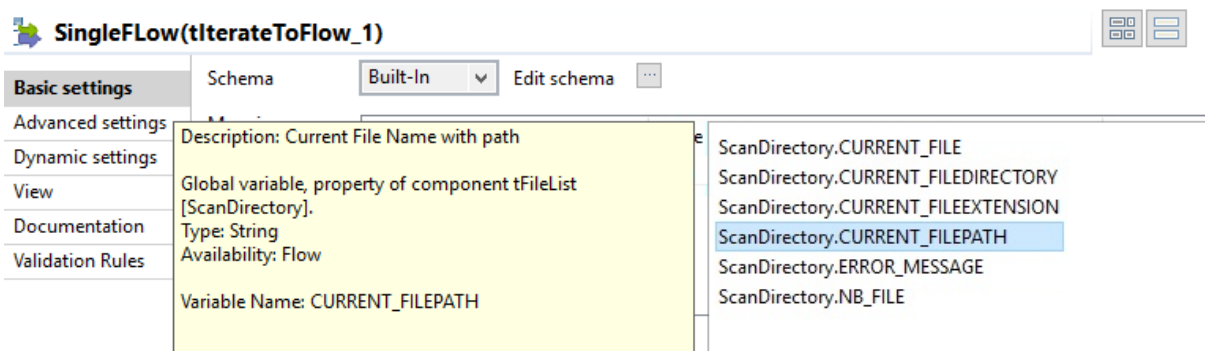
d.- Click **OK**.

3.- Configurar el valor de *FilePath*.

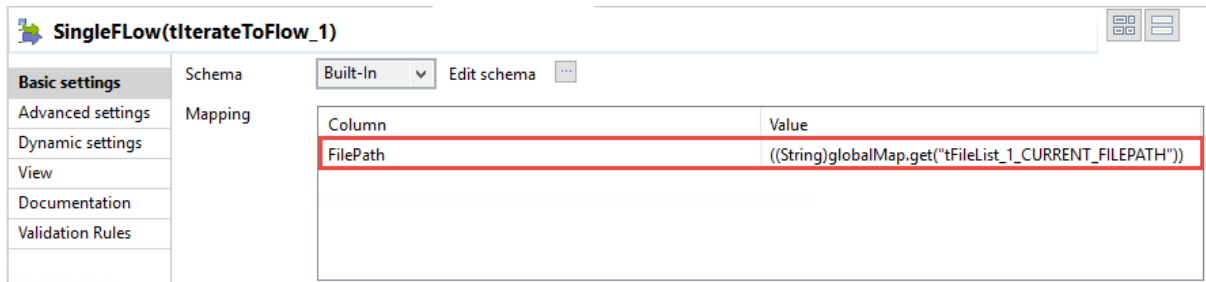
En la pestaña **Component**, la columna **FilePath** aparece en la tabla **Mapping**. Configure su valor utilizando la variable **CURRENT_FILEPATH**.

a.- En la tabla **Mapping**, a la derecha de la columna **FilePath**, click en **Value**.

b.- Comience a ingresar Scan y presione **CTRL+BARRA ESPACIADORA**. Las variables del componente aparecen en la lista de sugerencias.



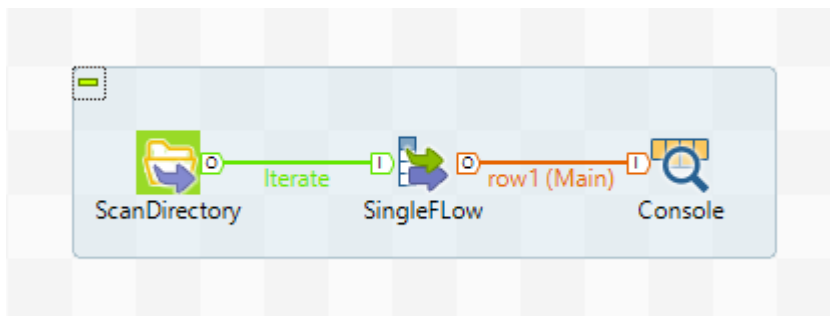
Seleccione **ScanDirectory.CURRENT_FILEPATH**.



4.- Use un componente tLogRow para mostrar la lista de archivos en pantalla.

a.- A la derecha de **SingleFlow**, coloque un componente **tLogRow** y nómbrelo **Console**.

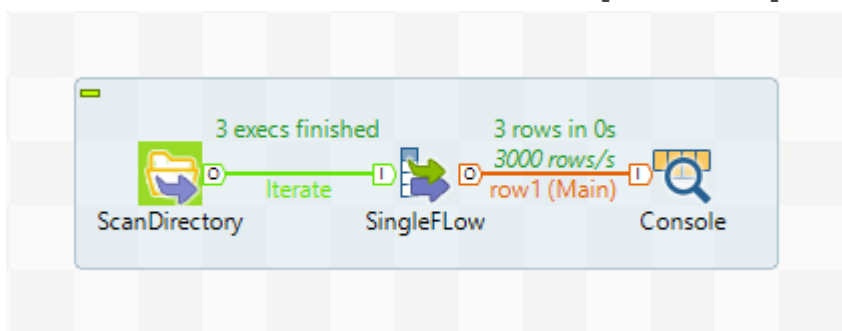
b.- Conecte **SingleFlow** con **Console** usando **Main** row.



5.- Para mostrar la lista de archivos en la consola, ejecute el trabajo.

a.- Click en la pestaña **Run**. Seleccione el context **Development**.

b.- Click **Run**. El Job debe finalizar en [exit code=0].



ScanDirectory genera 3 iteraciones:

Job listDirectory

- Basic Run
- Debug Run
- Advanced settings
- Target Exec
- Memory Run

Execution

```

Starting job listDirectory at 14:48 12/03/2021.
[statistics] connecting to socket on port 4054
[statistics] connected
C:\StudentFiles\DI\Basics\ExecutionControl\Staging\Shop1.csv
C:\StudentFiles\DI\Basics\ExecutionControl\Staging\Shop2.csv
C:\StudentFiles\DI\Basics\ExecutionControl\Staging\Shop3.csv
[statistics] disconnected

Job listDirectory ended at 14:48 12/03/2021. [Exit code = 0]

```

Development

Name	Value
DBControl_Additio...	noDateTimeString...
DBControl_Database	training
DBControl_Login	talend
DBControl_Password	****
DBControl_Port	3306
DBControl_Server	localhost
StagingDirectory	C:/StudentFiles/DI...

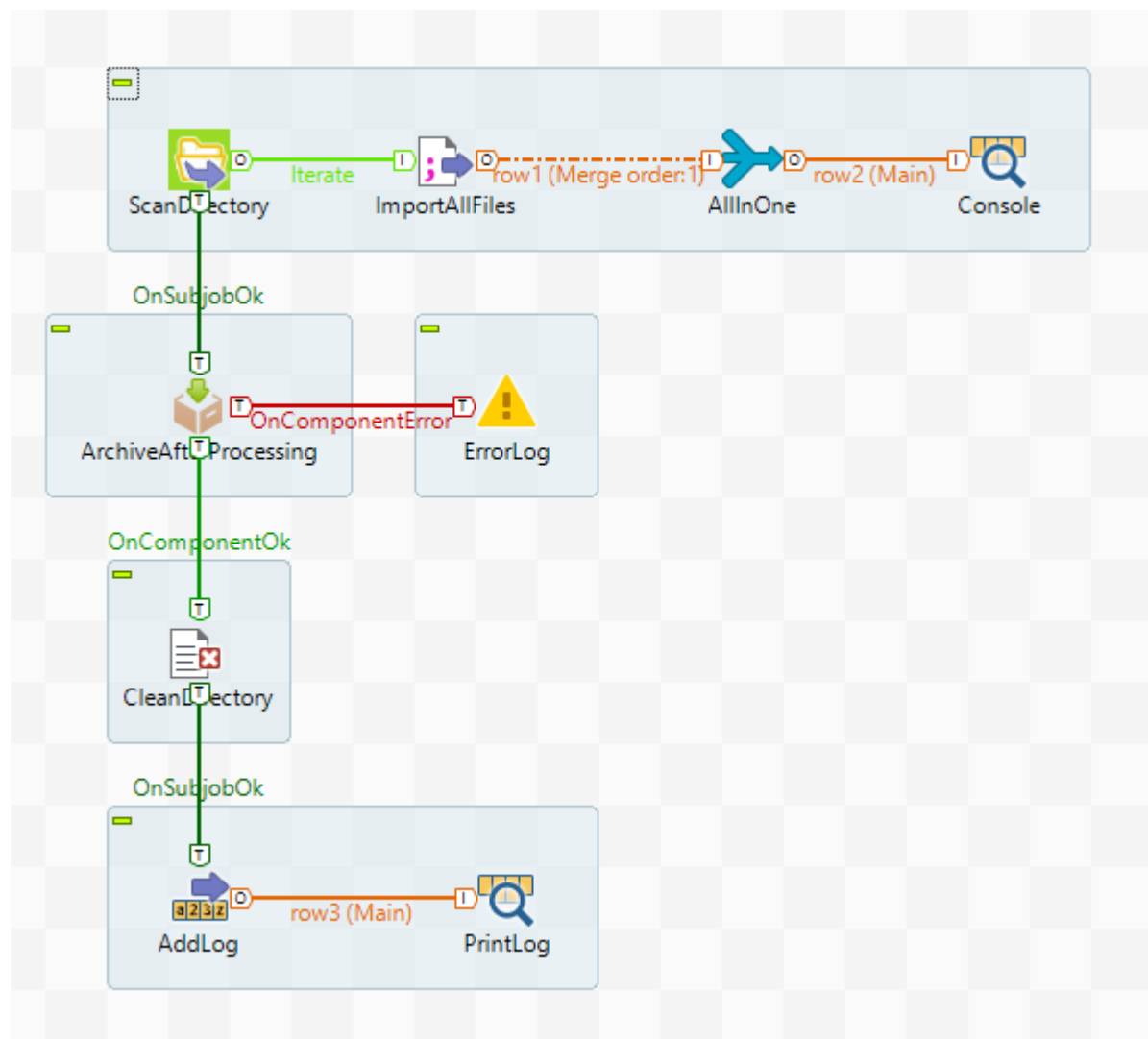
Las rutas de los tres archivos analizados en el directorio provisional se muestran en la consola.

Tema 3: Procesamiento de archivos

Antes de comenzar

En este ejercicio, procesará todos los archivos del directorio Staging. Si el Job finaliza sin errores, realiza una copia de seguridad y limpiará el directorio.

Al finalizar, el Job debe verse así:



Procesando todos los archivos

En este ejercicio, creará un Job nuevo y reutilizará el componente **ScanDirectory**. Finalmente, configurará el Job para procesar datos de todas las iteraciones en una sola salida.

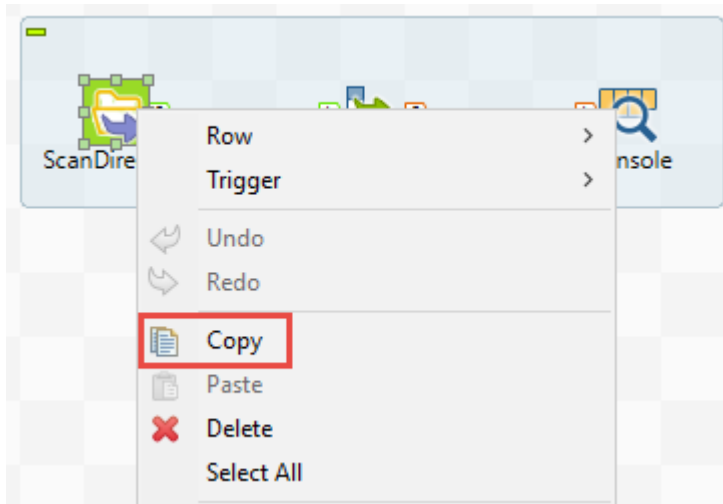
1.- Cree un nuevo Job named *processDirectory* e importe el grupo de contexto **ExecutionControl**. Luego copie el componente **ScanDirectory** de su Job anterior y péguelo en el **Designer**.

a.- Cree un nuevo standard Job en **Job Design > Standard > ExecutionControl** y nómbrelo *processDirectory*.

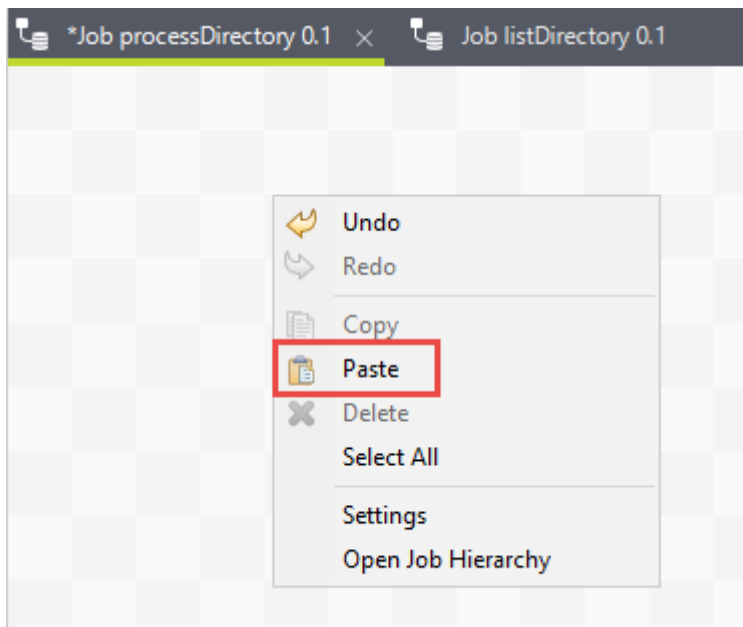
b.- Al igual que en el ejercicio anterior, en **Contexts**, importe el grupo de contexto **ExecutionControl** al nuevo Job.

	Name	Type	Comment	Production		Development	
				Value		Value	
1	ExecutionControl (from repository c						
2	DBControl_AdditionalParams	String		noDatetimeStringSync=true	<input type="checkbox"/>	noDatetimeStringSync=true	<input type="checkbox"/>
3	DBControl_Database	String		production	<input type="checkbox"/>	training	<input type="checkbox"/>
4	DBControl_Login	String		talend	<input type="checkbox"/>	talend	<input type="checkbox"/>
5	DBControl_Password	Password		*****	<input type="checkbox"/>	*****	<input type="checkbox"/>
6	DBControl_Port	String		3306	<input type="checkbox"/>	3306	<input type="checkbox"/>
7	DBControl_Server	String		localhost	<input type="checkbox"/>	localhost	<input type="checkbox"/>
8	StagingDirectory	Directory		C:/StudentFiles/DIBasics/ExecutionCo	<input type="checkbox"/>	C:/StudentFiles/DIBasics/ExecutionCo	<input type="checkbox"/>

c.- Abra el Job **listDirectory**. Haga click derecho sobre **ScanDirectory** y seleccione **Copy**.



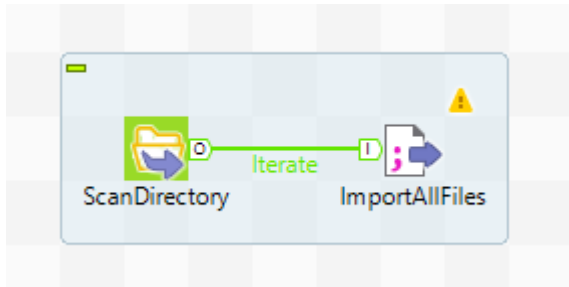
d.- Abra el Job **processDirectory**. Haga click derecho en **Designer** y seleccione **Paste**.



2.- Añada un componente **tFileInputDelimited** y configure su input path usando la variable **CURRENT_FILEPATH**.

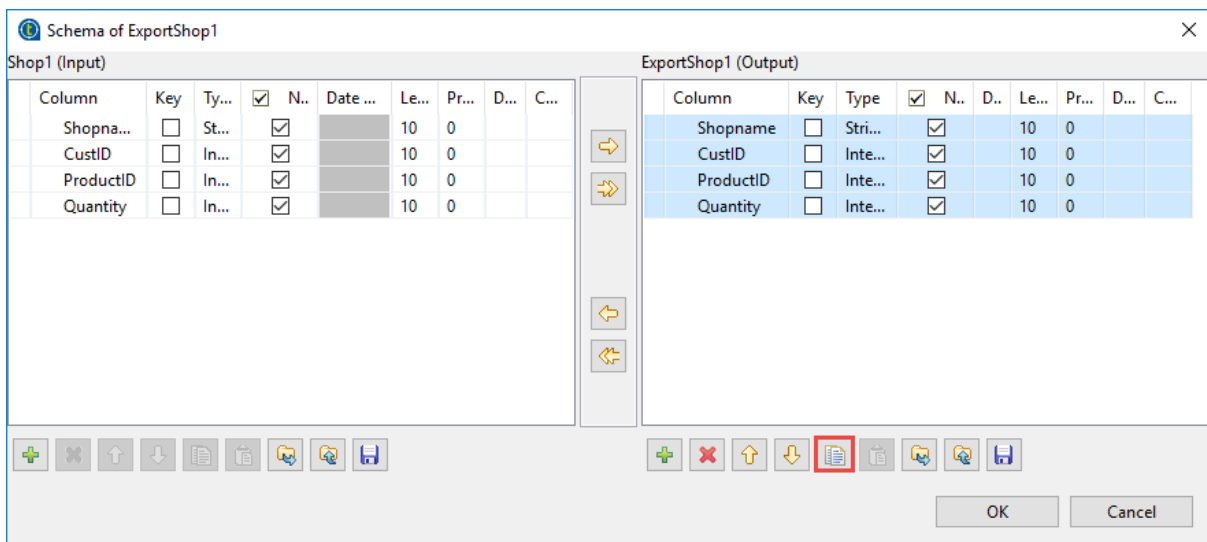
a.- En **Designer**, coloque un componente **tFileInputDelimited** a la derecha de **ScanDirectory**. Nómbrelo *ImportAllFiles*.

b.- Conecte **ScanDirectory** con **ImportAllFiles** usando **Iterate** row.



c.- Para configurar ImportAllFiles reutilizando el esquema de los archivos originales, abra el Job **filesStaging**.

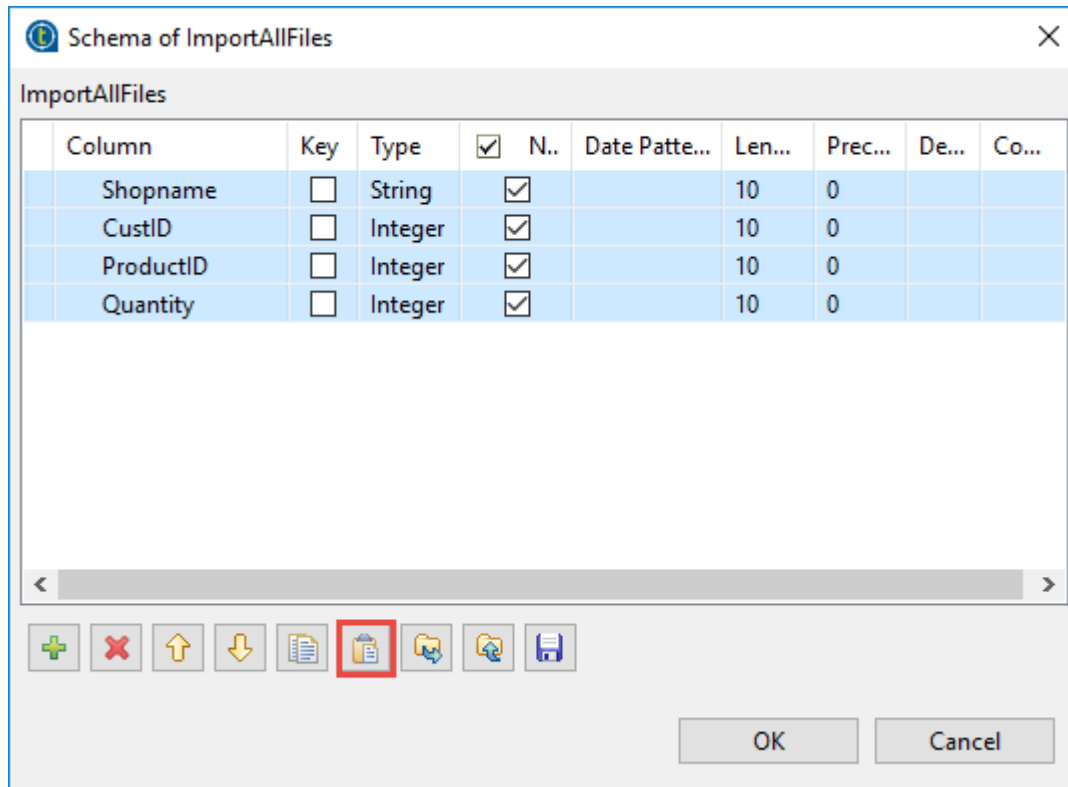
d.- Haga doble click en uno de los tres componentes **tFileOutputDelimited**, y en la pestaña **Component**, a la derecha de **Edit schema**, click [...].



A la derecha de la ventana, seleccione todas las filas para el esquema de salida y haga clic en el botón Copy, luego haga clic en OK.

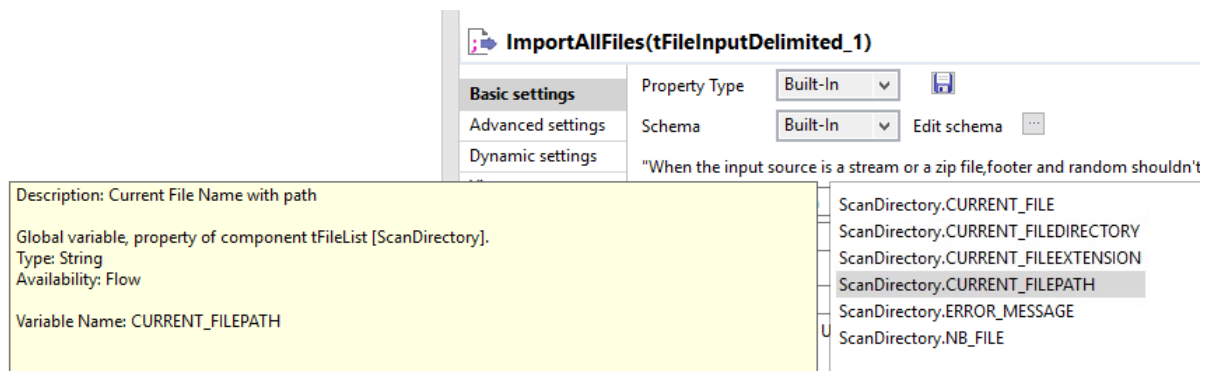
e.- Abra el Job **processDirectory** y haga doble click sobre **ImportAllFiles**.

f.- A la derecha de **Edit schema**, click [...].



Click en **Paste**, y luego en **OK**.

g.- Borre el contenido de **File name** y comience a escribir *Scan*, luego presione **CTRL+SPACEBAR**.



Seleccione **ScanDirectory.CURRENT_FILEPATH**.

h.- Para **Header**, ingrese 1.

ImportAllFiles(tFileInputDelimited_1)

Property Type: Built-In

Schema: Built-In Edit schema

"When the input source is a stream or a zip file, footer and random shouldn't be bigger than 0."

File name/Stream: `((String)globalMap.get("tFileList_1_CURRENT_FILEPATH"))`

Row Separator: `"\n"` Field Separator: `";"`

☐ CSV options

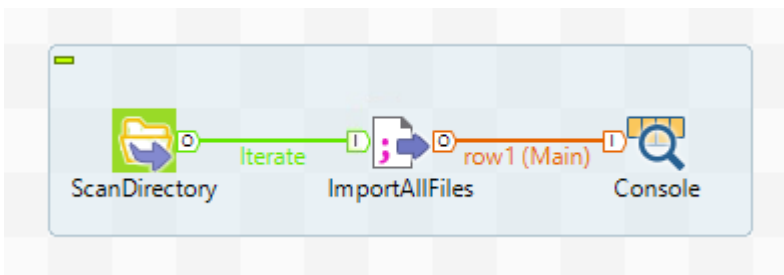
Header: **1** Footer: 0 Limit:

☒ Skip empty rows ☐ Uncompress as zip file ☐ Die on error

3.- Añada un componente tLogRow, luego ejecute el Job.

a.- A la derecha de **ImportAllFiles**, coloque un componente **tLogRow** y nómbrelo *Console*.

b.- Conecte **ImportAllFiles** con **Console** usando **Main** row.



c.- Abra la pestaña **Run**. Seleccione el contexto **Development**.

d.- Click **Run**.

El Job debe finalizar con **[exit code=0]**.

Job processDirectory

Basic Run

Execution

Run Kill Clear

```

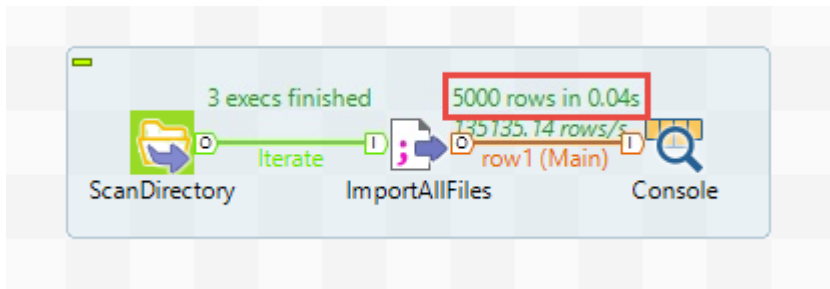
Shop3|56|56|508
Shop3|53|47|211
Shop3|73|30|22
Shop3|8|34|156
Shop3|37|49|752
Shop3|89|6|753
Shop3|86|52|145
[statistics] disconnected

Job processDirectory ended at 16:15 12/03/2021. [Exit code = 0]
  
```

Development

Name	Value
DBControl_Additio...	noDatetimeString...
DBControl_Database	training
DBControl_Login	talend
DBControl_Password	****
DBControl_Port	3306
DBControl_Server	localhost
StagingDirectory	C:/StudentFiles/DI...

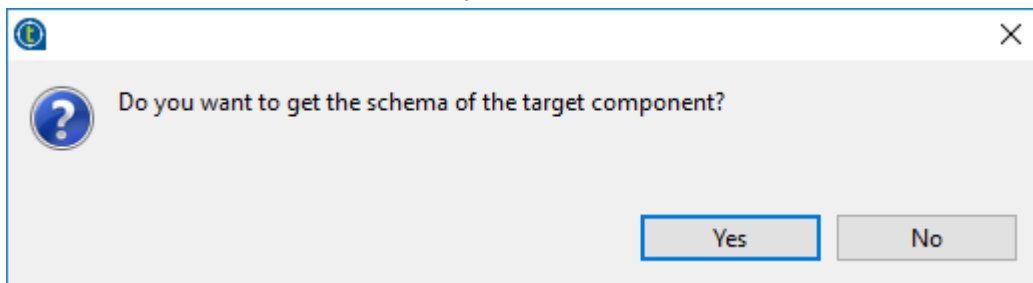
Como se muestra en Designer, el contenido de un solo archivo (5000 rows) se ha enviado a tLogRow.



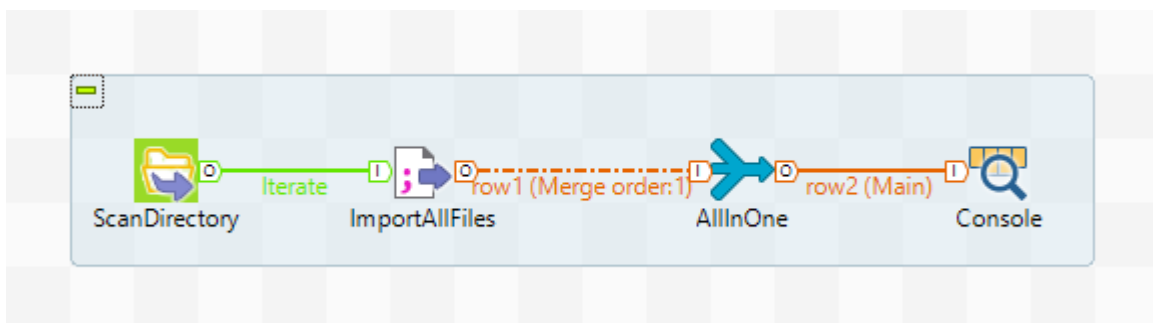
4.- Para recopilar todos los datos extraídos de los archivos de origen, inserte un componente tUnite.

a.- En **Designer**, coloque un componente **tUnite** a la izquierda de **Console** y nómbrelo *AllInOne*.

b.- Conecte **ImportAllFiles** y **Console** usando **Main** rows.



Cuando se le pregunte si desea obtener el esquema del componente de destino, haga click en **Yes**.



5.- Ejecute el Job de nuevo y observe el número de rows que se muestra por consola.

a.- Abra la ventana **Run**. Seleccione el contexto **Development**.

b.- Click **Run**

El Job debe finalizar con **[exit code=0]**.

Job processDirectory

- Basic Run
- Debug Run
- Advanced settings
- Target Exec
- Memory Run

Execution

Run Kill Clear

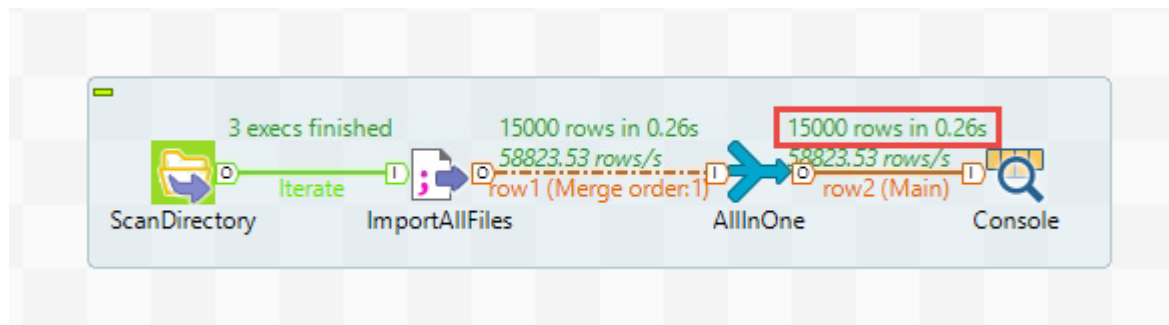
Shop3|56|56|508
Shop3|53|47|211
Shop3|73|30|22
Shop3|8|34|156
Shop3|37|49|752
Shop3|89|6|753
Shop3|86|52|145
[statistics] disconnected

Job processDirectory ended at 16:15 12/03/2021. [Exit code = 0]

Development

Name	Value
DBControl_Additio...	noDatetimeString...
DBControl_Database	training
DBControl_Login	talend
DBControl_Password	****
DBControl_Port	3306
DBControl_Server	localhost
StagingDirectory	C:/StudentFiles/DI...

En esta ocasión, el contenido de los tres archivos (un total de 15.000 filas) se ha enviado a tLogRow.

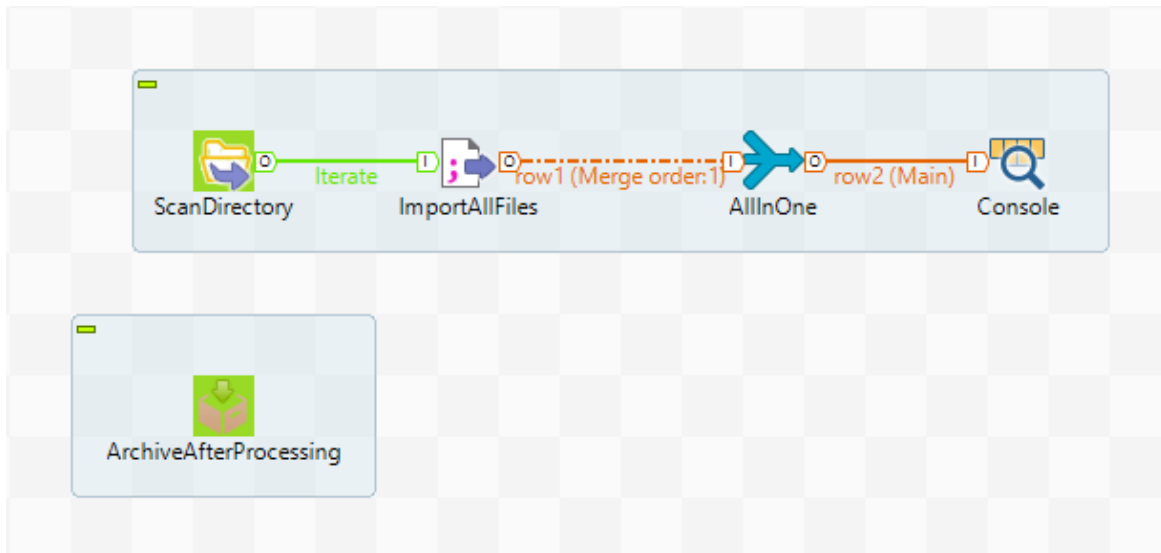


Usar triggers

En este ejercicio, agregará más componentes y los conectará mediante triggers para ejecutarlos en diferentes condiciones

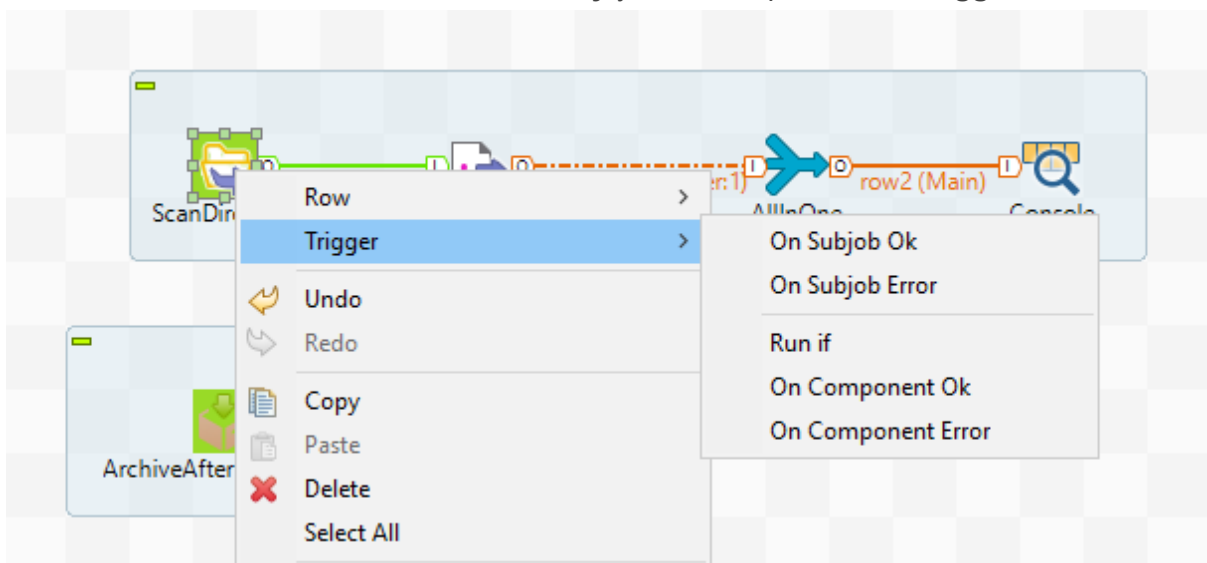
1.- Use un componente **tFileArchive** para comprimir los archivos de origen solo si el subJob finaliza correctamente.

a.- Debajo de **ScanDirectory**, coloque un componente **tFileArchive** y nómbrelo *ArchiveAfterProcessing*.

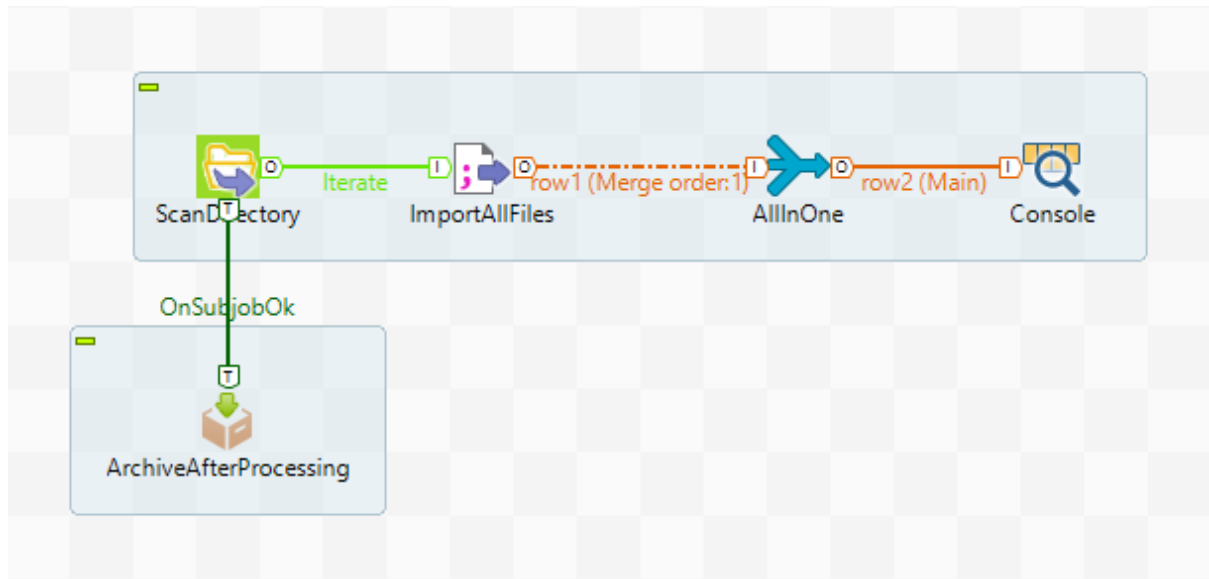


💡 Para seguir las mejores prácticas de diseño, organice los subJobs para que fluyan de arriba a abajo.

b.- Click derecho sobre **ScanDirectory** y abra las opciones de **Trigger**.



c.- Seleccione **On Subjob Ok**, luego presione **ArchiveAfterProcessing**.



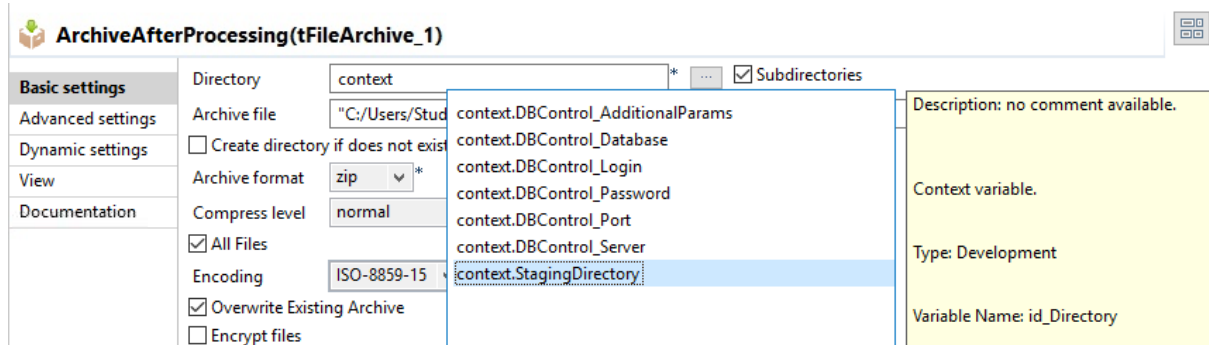
💡 El subJob que contiene el componente tFileArchive se ejecuta solo si el primer subJob finaliza sin errores.

2.- Configure tFileArchive.

Configure **ArchiveAfterProcessing** para comprimir el contenido del directorio Staging en un archivo de almacenamiento con una marca de tiempo en su nombre

a.- Haga doble click en **ArchiveAfterProcessing**.

b.- Borre el contenido de **Directory** y comience a ingresar *context*, luego presione **CTRL+SPACEBAR**.



Del listado de variables, seleccione **context.StagingDirectory**.

c.- En **Archive file**, ingrese "C:/Backup/archive_" luego añada +getdate y presione **CTRL+SPACEBAR**.

Directory	context.StagingDirectory *	<input checked="" type="checkbox"/> Subdirectories
Archive file	"C:/Backup/archive_" + getdate	
	getDate(String pattern) : id_String - TalendDate	

Seleccione la función **getDate**, actualice el formato de la fecha, y complete la ruta de la siguiente manera:

"C:/Backup/archive_" + TalendDate.getDate("CCYYMMDD_hhmmss") + ".zip"

d.- Seleccione **Create directory if does not exist**.

3.- Ejecute el Job y compruebe que se haya creado el archivo.

a.- Abra la pestaña **Run**. Seleccione el contexto **Development**.

b.- Click **Run**. El Job debe terminar con **[exit code=0]**.

c.- Busque en **C:\Backup**

This PC > Local Disk (C:) > Backup				
Search Backup				
Name	Date modified	Type	Size	
archive_20210315_130616	3/15/2021 1:06 PM	Compressed (zipp...	75 KB	

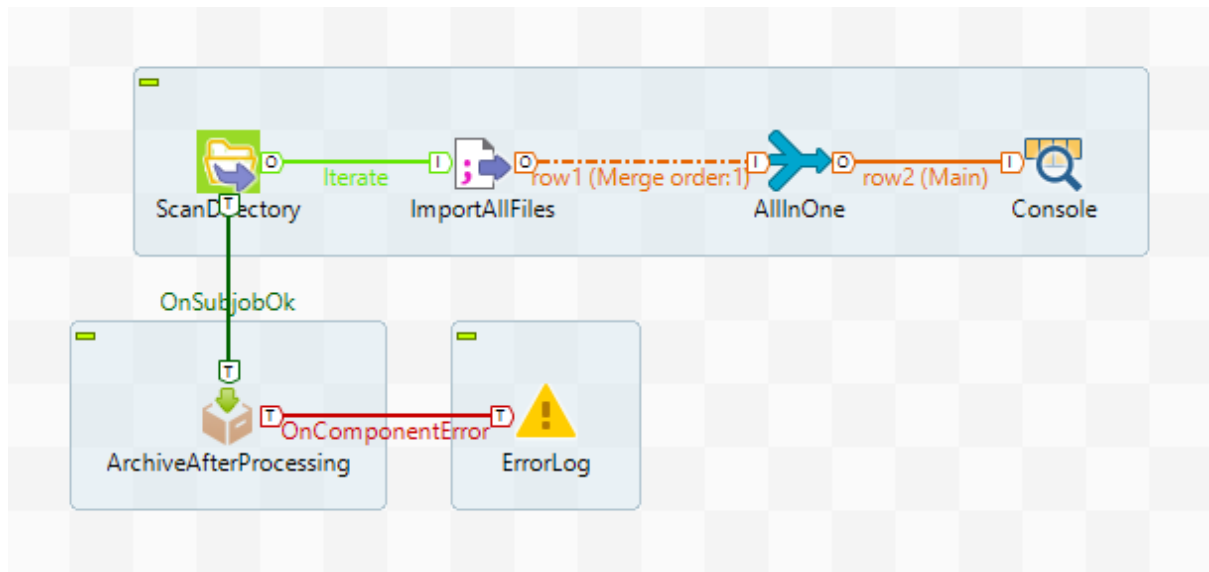
El archivo se creó con éxito.

4.- Añada un componente **tWarn** para mostrar un mensaje en pantalla si la compresión falla.

a.- Coloque un componente **tWarn** a la derecha de **ArchiveAfterProcessing** y

nómbrelo *ErrorLog*.

b.- Conecte **ArchiveAfterProcessing** con **ErrorLog** usando un trigger **On Component Error**.



💡 El subJob se ejecutará sólo si el componente ArchiveAfterprocessing no genera un error.

c.- Doble click en **ErrorLog**.

d.- En **Warn message** ingrese: *"this is a warning!"*
" + ((Integer)globalMap.get("tFileList_1_NB_FILE")) + " files were listed in "
+ context.StagingDirectory + " but the compression process failed with the error message:
" + ((String)globalMap.get("tFileArchive_1_ERROR_MESSAGE"))

ErrorLog(tWarn_1)

Basic settings	Warn message	"this is a warning! " + ((Integer)globalMap.get("tFileList_1_NB_FILE")) + " files were listed in " + context.StagingDirec
Advanced settings	Code	42
Dynamic settings	Priority	Warning
View		
Documentation		

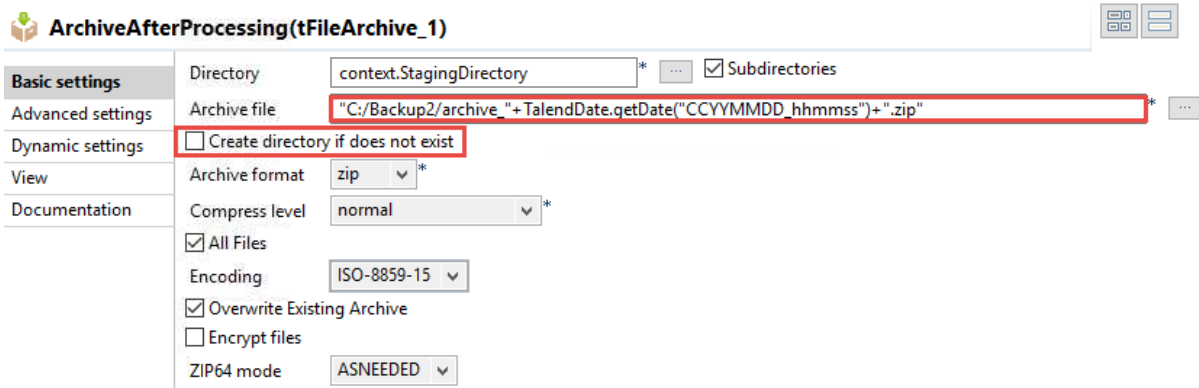
💡 Este mensaje combina variables de contexto y de componente.

5.- Provoque un error y ejecute el Job para comprobar el funcionamiento.

a.- Haga doble click en **ArchiveAfterProcessing** e ingrese estos parámetros:

→ Desmarque la opción **Create directory if does not exist**.

→ Cambie la ruta del archivo **Archive file** añadiendo 2 al lado de **Backup**.



ArchiveAfterProcessing(tFileArchive_1)

Basic settings

Directory: context.StagingDirectory * ☒ Subdirectories

Advanced settings

Archive file: "C:/Backup2/archive_" + TalendDate.getDate("CCYYMMDD_hhmmss") + ".zip" *

Dynamic settings

☒ Create directory if does not exist

View

Archive format: zip *

Documentation

Compress level: normal *

☒ All Files

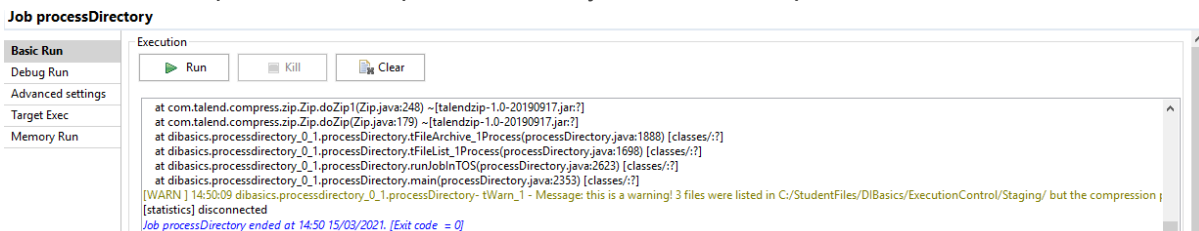
Encoding: ISO-8859-15

☒ Overwrite Existing Archive

☐ Encrypt files

ZIP64 mode: ASNEEDED

b.- En la pestaña **Run**, presione **Run** y observe la respuesta.



Job processDirectory

Basic Run

Execution

at com.talend.compress.zip.Zip.doZip1(Zip.java:248) ~[talendzip-1.0-20190917.jar:?]
 at com.talend.compress.zip.Zip.doZip(Zip.java:179) ~[talendzip-1.0-20190917.jar:?]
 at dibasics.processdirectory_0_1.processDirectory.tFileArchive_1Process(processDirectory.java:1888) [classes:/?]
 at dibasics.processdirectory_0_1.processDirectory.tFileList_1Process(processDirectory.java:1698) [classes:/?]
 at dibasics.processdirectory_0_1.processDirectory.runJobInTOS(processDirectory.java:2623) [classes:/?]
 at dibasics.processdirectory_0_1.processDirectory.main(processDirectory.java:2353) [classes:/?]
 [WARN] 14:50:09 dibasics.processdirectory_0_1.processDirectory- tWarn_1 - Message: this is a warning! 3 files were listed in C:/StudentFiles/DIBasics/ExecutionControl/Staging/ but the compression statistics disconnected
 Job processDirectory ended at 14:50 15/03/2021. [Exit code = 0]

El mensaje de advertencia se muestra en verde.

c.- Antes de continuar, revierta los cambios en **ArchiveAfterProcessing**:

- Seleccione **Create directory if does not exist**.
- Remueva el 2 en la ruta del archivo en **Archive file**.

Borrar una carpeta

En este ejercicio, agregará más componentes para borrar la carpeta y mostrar mensajes en el log, y los conectará mediante triggers.

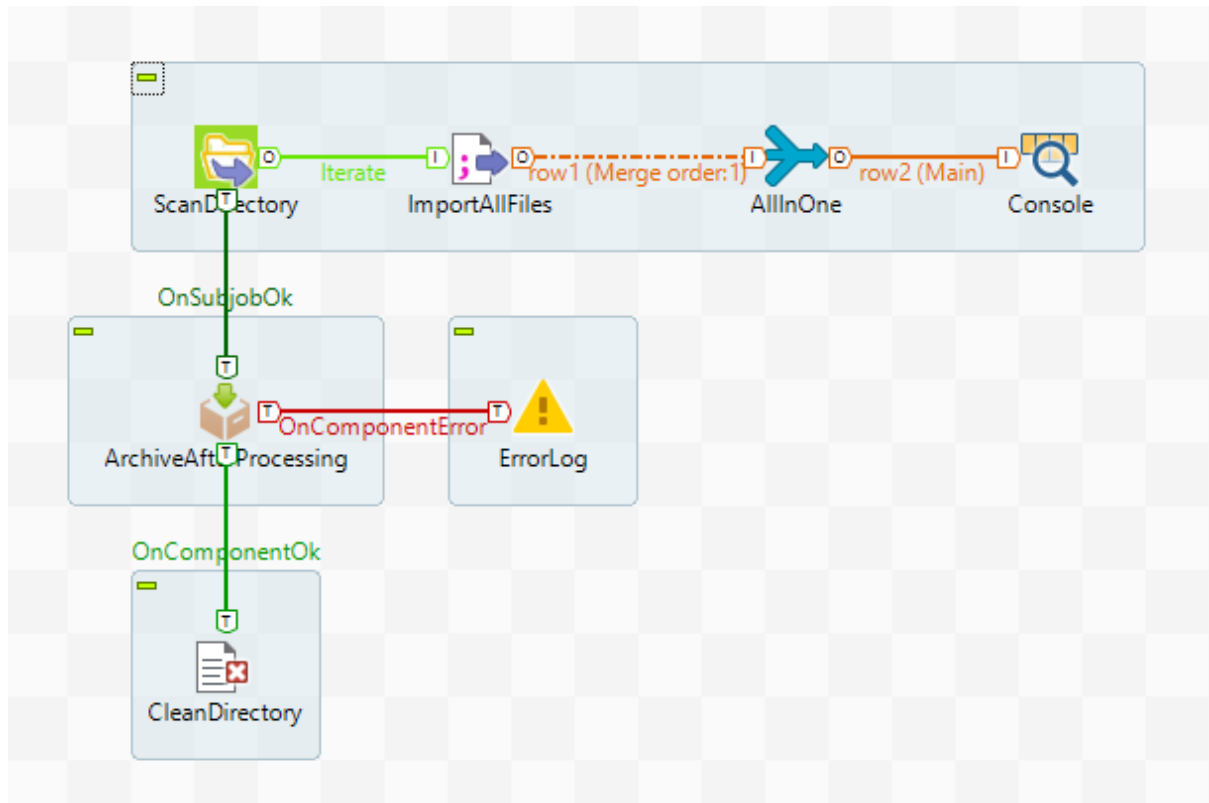
1.- Borrar una carpeta.

Añada un componente **tFileDelete**, que se ejecutará sólo si **ArchiveAfterProcessing** finaliza correctamente.

a.- Debajo de **ArchiveAfterProcessing**, coloque un componente **tFileDelete** y nómbrelo *CleanDirectory*.

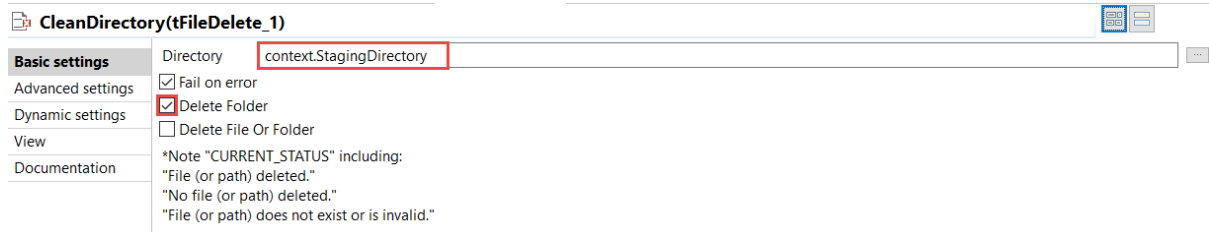
b.- Conecte **ArchiveAfterProcessing** con **CleanDirectory** usando un trigger **On**

Component Ok.



c.- Haga doble click en **CleanDirectory** y seleccione estos parámetros:

- Seleccione **Delete Folder**.
- En **Directory**, ingrese *context.StagingDirectory*.

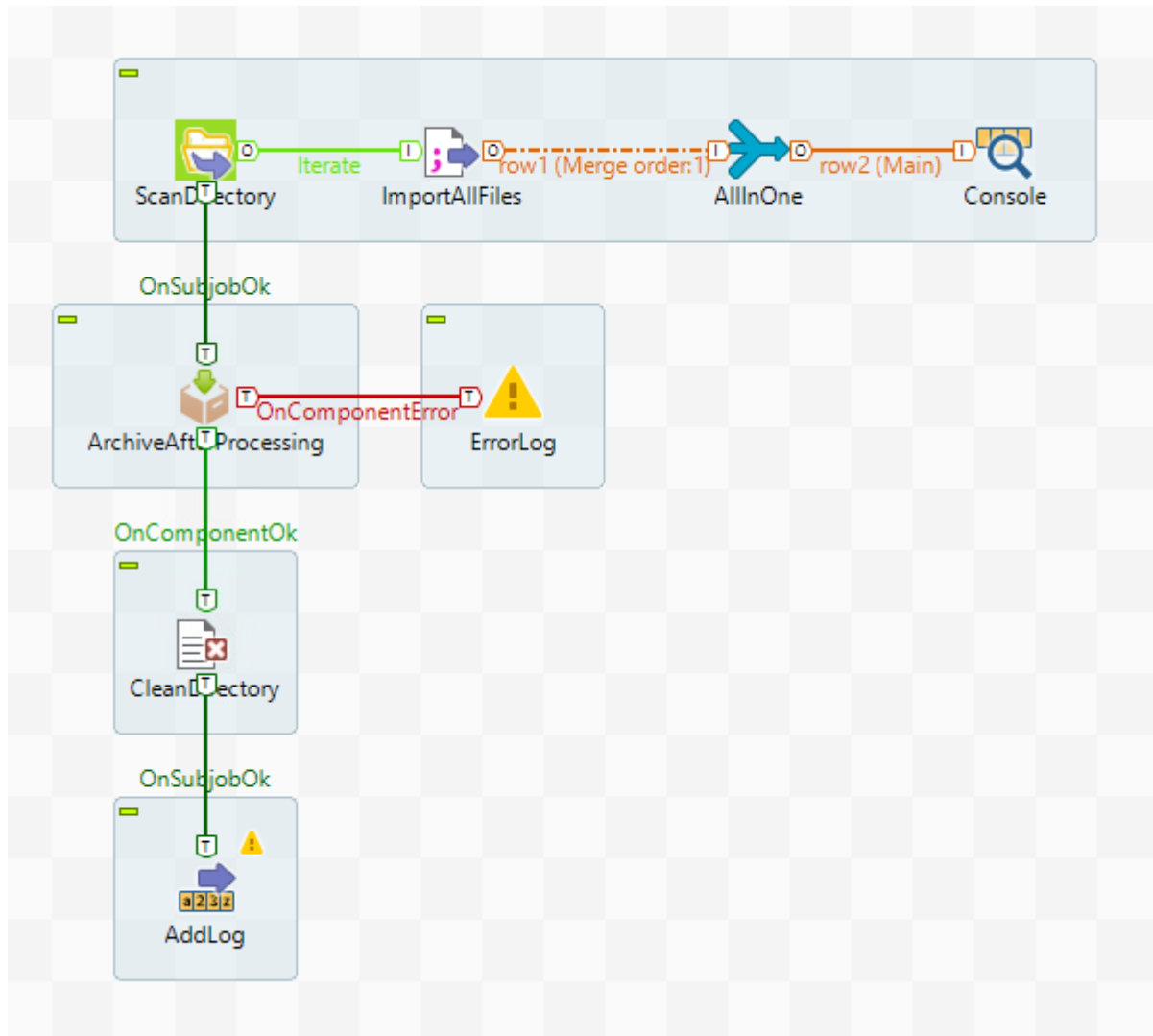


2.- Confirme que se eliminó la carpeta.

Añada un componente **tFixedFlowInput** para crear un mensaje log one-column reusando la variable **DELETE_PATH** del componente **tFileDelete**. Este mensaje se debe mostrar en pantalla sólo si se borró la carpeta en el subJob anterior.

a.- Debajo de **CleanDirectory**, coloque un componente **tFixedFlowInput** y nómbrelo *AddLog*.

b.- Conecte **CleanDirectory** con **AddLog** usando un trigger **On Subjob Ok**.












c.- Haga doble click en **AddLog**, a la derecha de **Edit schema**, click [...].

En la barra de herramientas, click **[+]** y cree una nueva columna. Nómbrala **Log**, y para **Type**, seleccione **String**.

Schema of AddLog

AddLog

Column	Key	Type	<input checked="" type="checkbox"/>	N..	D..	Len...	Prec...	De...	Co...
Log	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>						

OK Cancel

Click **OK**.

d.- En la tabla **Values**, ingrese el valor para la columna **Log**: *"Directory "+((String)globalMap.get("tFileDelete_1_DELETE_PATH"))+" has been deleted"*

AddLog(tFixedFlowInput_1)

Basic settings

Advanced settings

Dynamic settings

View

Documentation

Validation Rules

Schema Built-In Edit schema

Number of rows 1

Mode

- Use Single Table

Values

Column	Value
Log	"Directory "+((String)globalMap.get("tFileDelete_1_DELETE_PATH"))+" has been deleted"

Use Inline Table

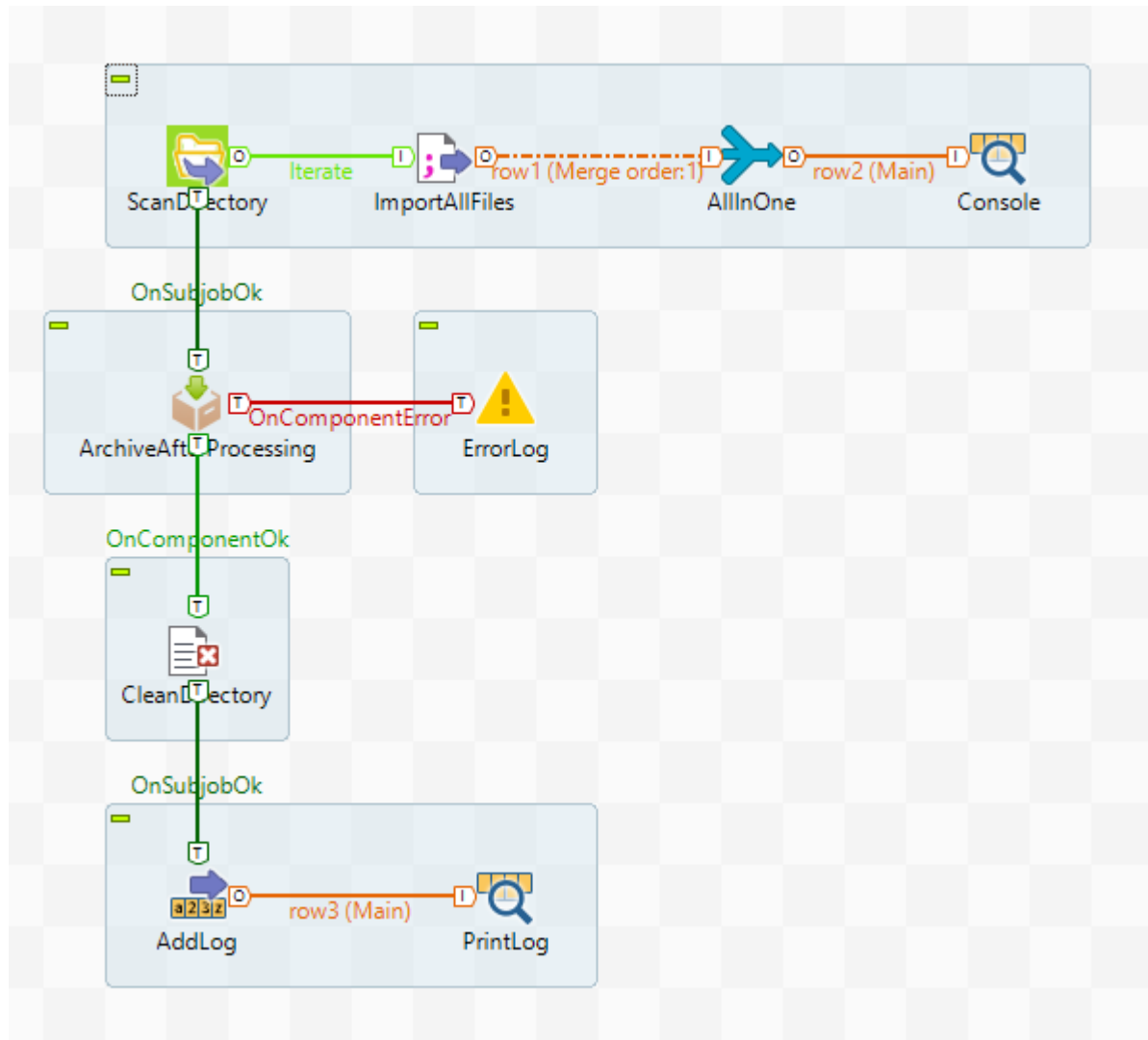
Use Inline Content(delimited file)

💡 Este mensaje reutiliza la variable DELETE_PATH del componente tFileDelete.

3.- Añada un componente **tLogRow** para imprimir el mensaje por consola.

a.- A la derecha de **AddLog**, coloque un componente **tLogRow** y nómbrelo *PrintLog*.

b.- Conecte AddLog con PrintLog usando Main row.



4.- Ejecute el Job y confirme que la carpeta **Staging** haya sido eliminada.

a.- Abra la pestaña **Run**. Seleccione el contexto **Development**.

b.- Click **Run**. El Job debe finalizar con **[exit code=0]** y mostrar el mensaje en pantalla.

Job processDirectory

Basic Run

Debug Run

Advanced settings

Target Exec

Memory Run

Execution

Run Kill Clear

```

Shop3|98|55|763
Shop3|84|30|826
Shop3|56|56|508
Shop3|53|47|211
Shop3|73|30|22
Shop3|8|34|156
Shop3|37|49|752
Shop3|89|6|753
Shop3|86|52|145
Directory C:/StudentFiles/DIBasics/ExecutionControl/Staging/ has been deleted
[statistics] disconnected
Job processDirectory ended at 06:25 27/11/2018. [exit code=0]

```

Line limit 100 Wrap

c.- Navegue hasta *C:\StudentFiles\DI\Basics\ExecutionControl* y confirme que la carpeta **Staging** ha sido eliminada.

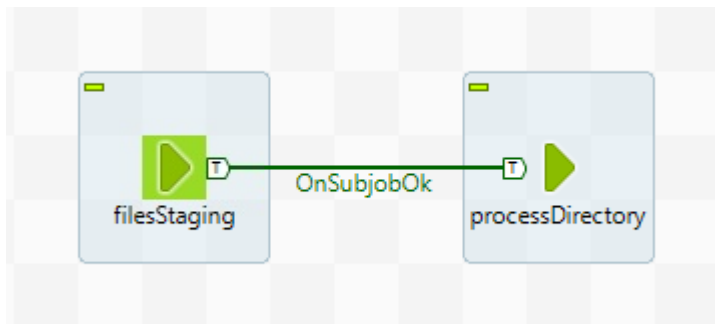
💡 Puede recrear la ruta y los tres archivos de origen automáticamente ejecutando el Job fileStaging.

Tema 4: Administrar la ejecución de un Job usando un master Job

Antes de comenzar

En este ejercicio, creará un master Job para ejecutar los Jobs fileStaging y processDirectory. Aprenderá a configurar las variables del Job en el master Job. Luego, exportará el master Job con sus dependencias, incluidas las variables de contexto y los metadatos.

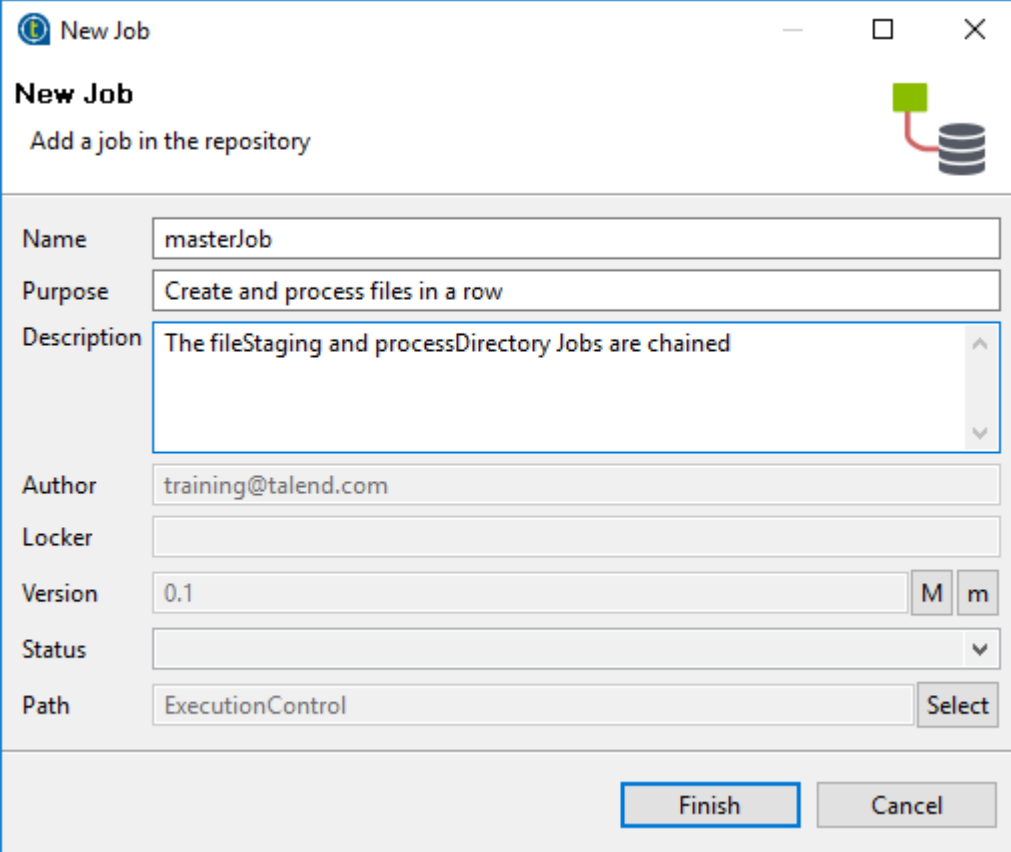
Al finalizar, su Job debería verse así:



Crear un master Job

In this exercise, you create a new Job and chain it to the two other Jobs in the Designer using tRunJob components. Then you run the Job and check the results.

1.- Cree un nuevo standard Job en **Job Designs > Standard > ExecutionControl** y nómbrelo *masterJob*.



New Job
Add a job in the repository

Name: masterJob

Purpose: Create and process files in a row

Description: The fileStaging and processDirectory Jobs are chained

Author: training@talend.com

Locker:

Version: 0.1 M m

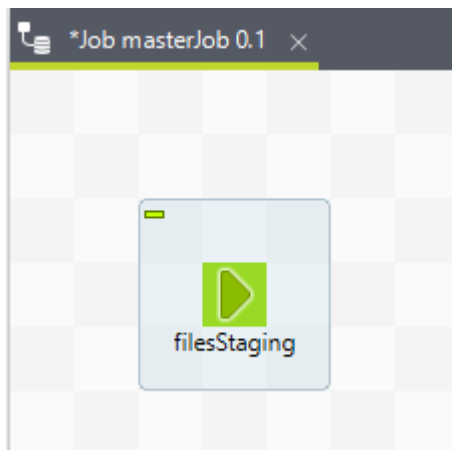
Status:

Path: ExecutionControl Select

Finish Cancel

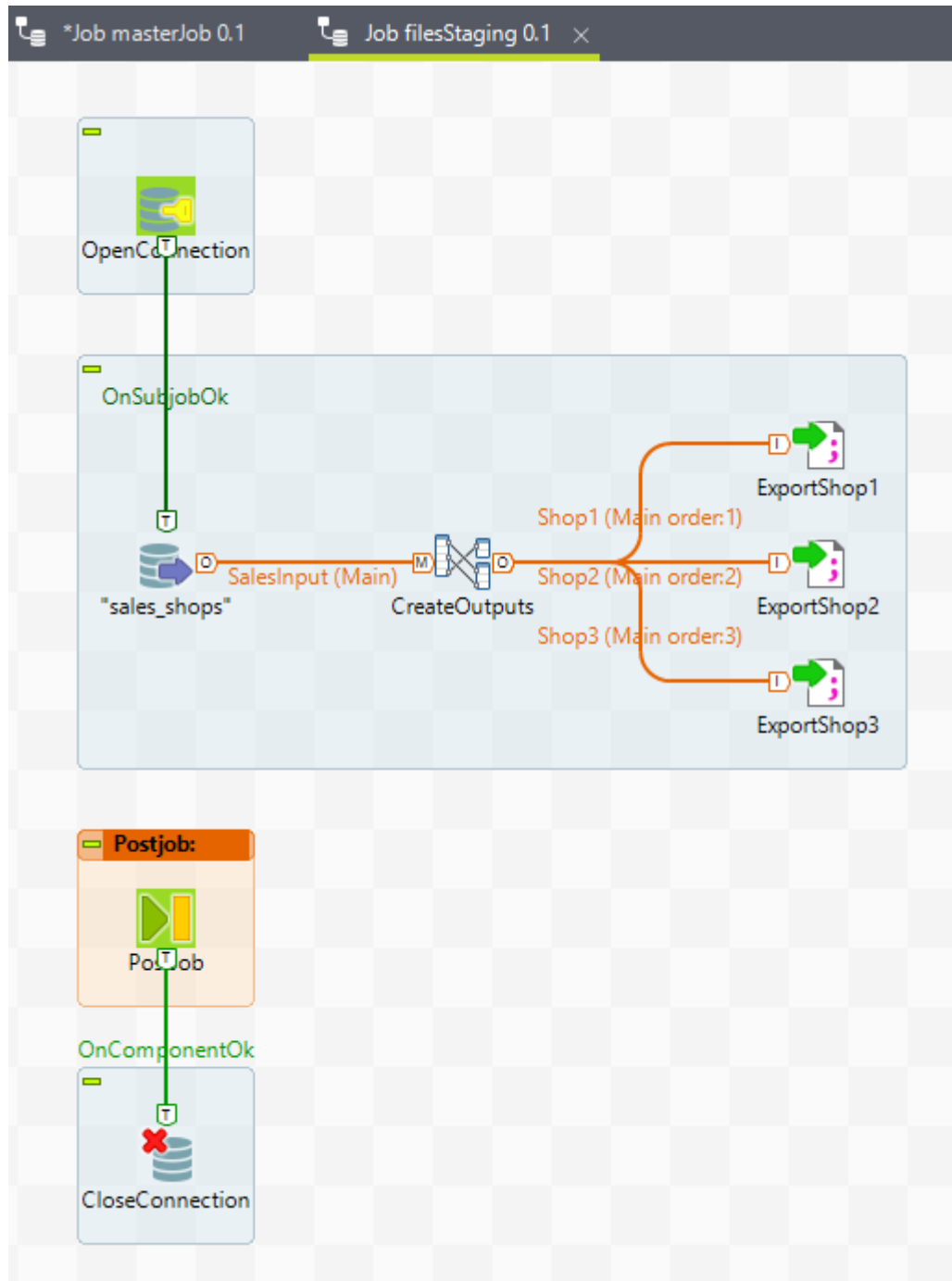
2.- Añadir un Job.

a.- Arraste el Job **filesStaging** desde el **Repository** al **Designer**. Esto agrega un componente tRunJob configurado para ejecutar el Job.



b.- Haga doble click en **filesStaging**. Normalmente, al hacer doble clic en un componente se abre la pestaña Component. Sin embargo, con los componentes tRunJob,

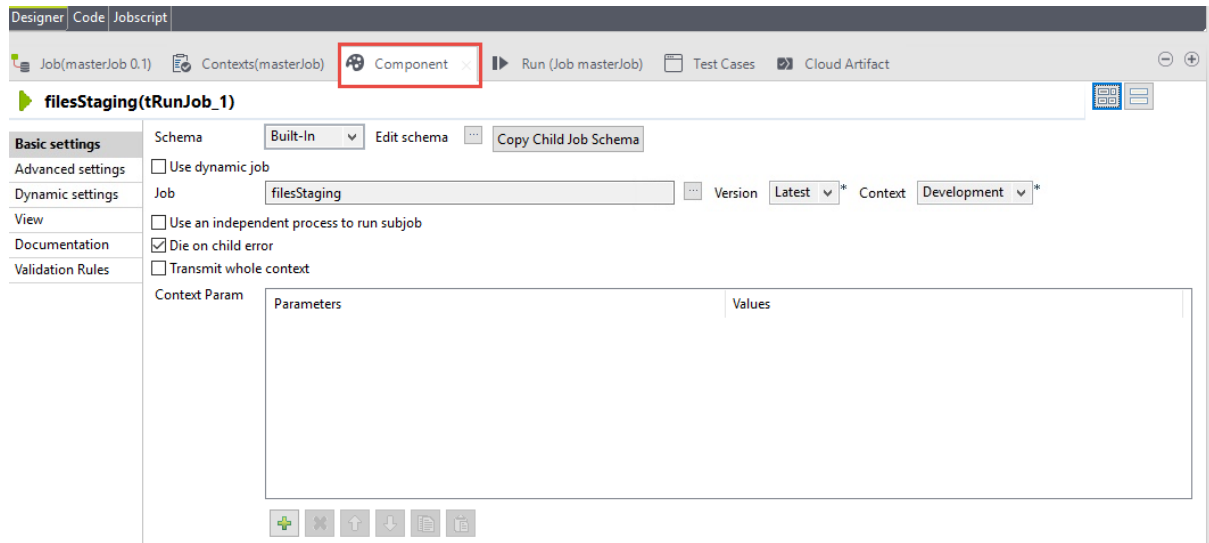
un doble clic abre el Job al que se hace referencia en una nueva pestaña **Designer**.



c.- Vuelva al **masterJob**. Click en **fileStaging** para seleccionarlo, luego abra la pestaña **Component** para ver la configuración.

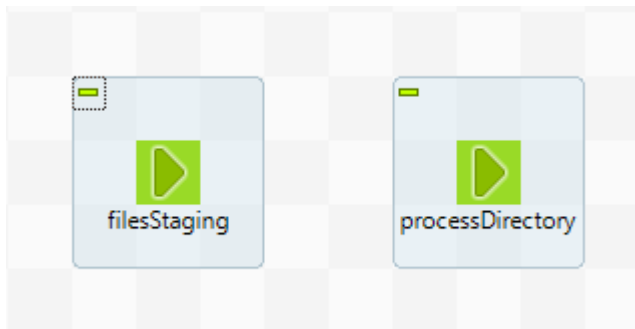
Puede ver las opciones para administrar la versión del Job que se ejecutará, así como el contexto en el que ejecutar el Job. Los parámetros de contexto individuales también son

configurables.

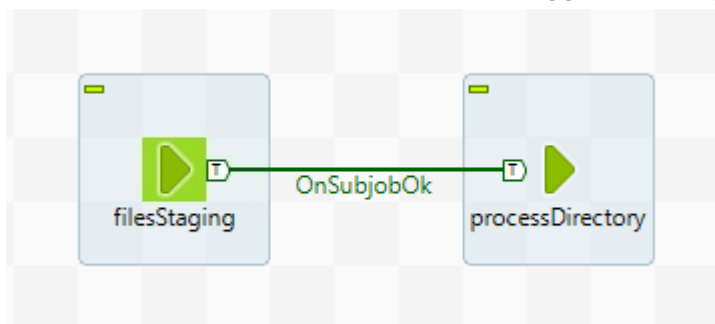


3.- Usando el mismo procedimiento, añada el Job **processDirectory** y conéctelo con **fileStaging**.

a.- Arraste el Job **processDirectory** desde el **Repository** al **Designer**.



b.- Conecte los Jobs usando un trigger **On Subjob OK**.

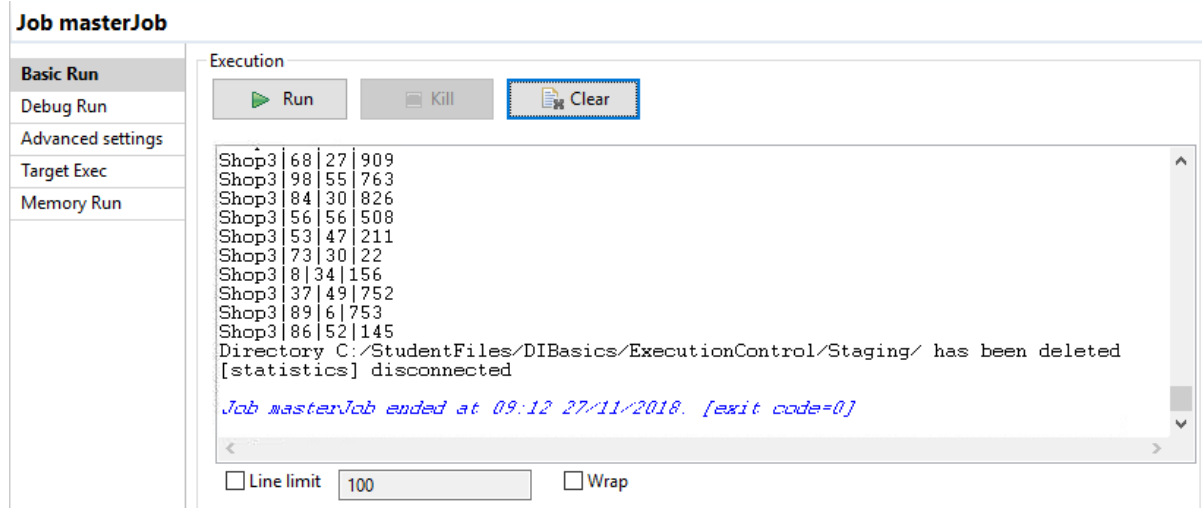


💡 Al conectar Jobs mediante componentes tRunJob, una buena práctica es vincularlos usando **On Subjob Ok** (en lugar de **On Component Ok**).

4.- Ejecute el master Job y confirme que se ha creado un nuevo archivo.

a.- Abra la pestaña **Run**.

b.- Click **Run**. Confirme que el Job finaliza con **[exit code=0]** y que el mensaje se ha mostrado en pantalla.



c.- Navegue a **C:\Backup**

Confirme que se ha generado un nuevo archivo.

💡 Debido a que ha probado cada Job de forma independiente, su master Job debería ejecutarse sin errores.

Transmitir variables de contexto entre Jobs

Para ejecutar el master Job, Studio usa los valores de las variables de contexto configurados en los Jobs originales. Esos valores se pueden cambiar en cada instancia de tRunJob.

Recuerde que usó variables de contexto para configurar la ruta de staging. En este ejercicio, actualizará estas variables usando dos métodos diferentes. Primero, transmitirá el contexto del master Job a los Jobs originales. Luego configurará el valor de la variable directamente desde la configuración de tRunJob. Cuando se ejecute el Jobs, gracias al mensaje que se muestra en la consola, puede verificar la ruta.

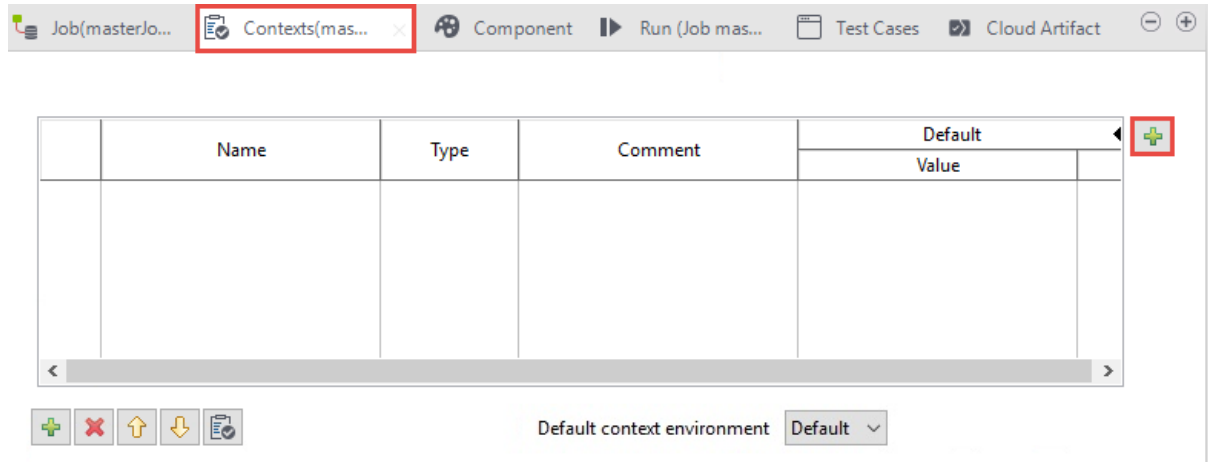
Sin embargo, antes de eso, debe configurar los contextos de Development y Production para el master Job.

💡 El Job fileStaging usa la variable StagingDirectory para determinar la ruta de los archivos que crea, y el Job processDirectory la usa para determinar la ruta del directorio a procesar.

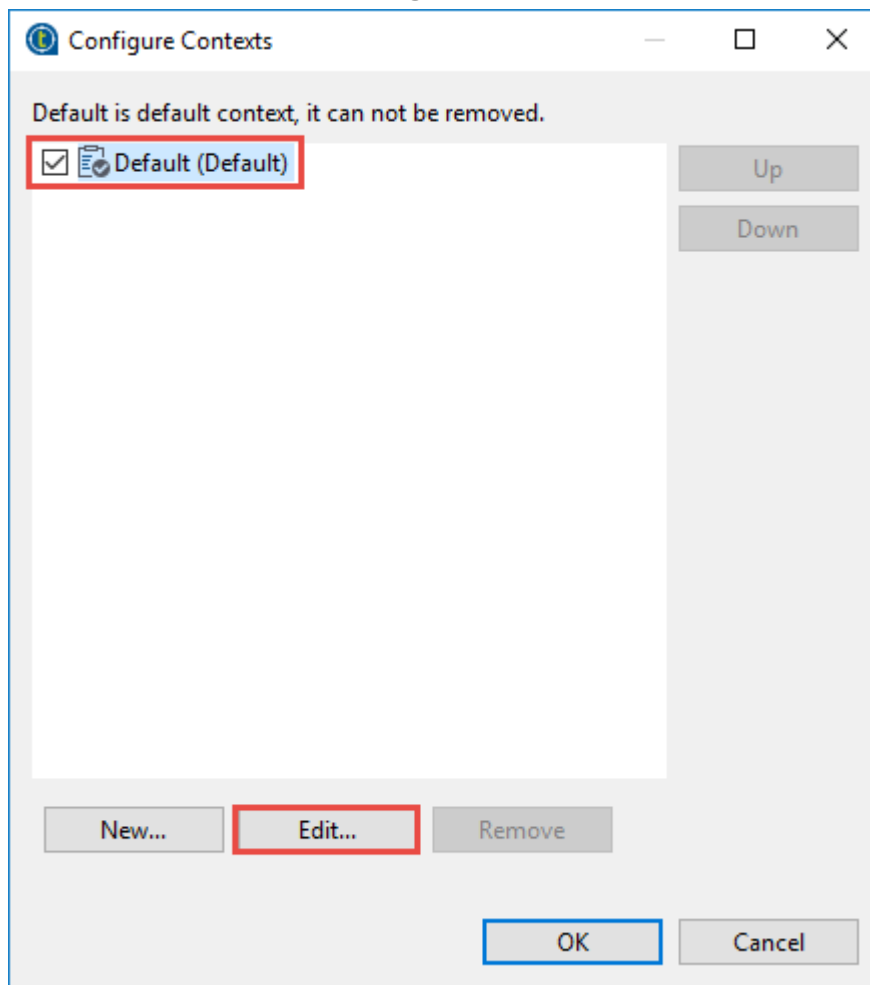
Asegúrese de configurar esta variable de la misma manera para ambos componentes de tRunJob.

1.- Configure el contexto por defecto y renómbrelo.

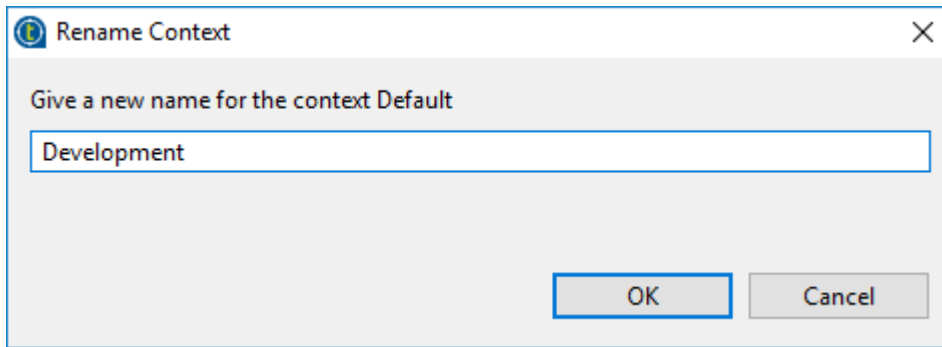
a.- En la pestaña **Contexts**, a la derecha de la tabla, click [+].



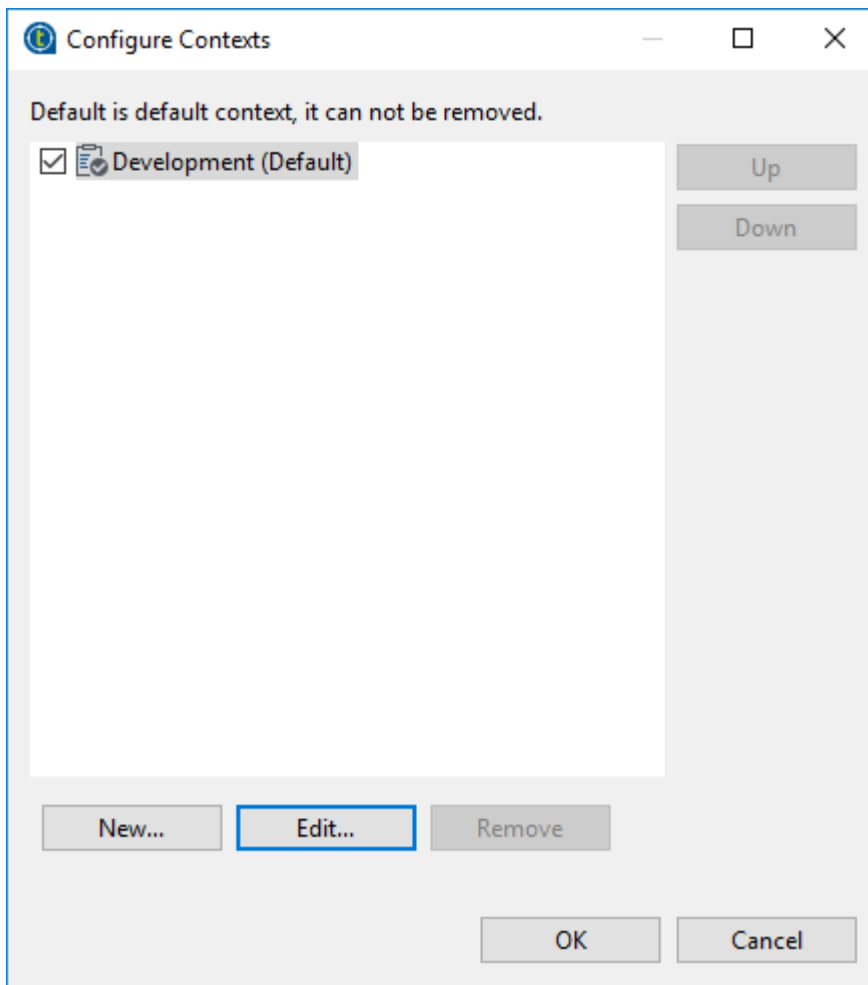
b.- En la ventana **Configure Contexts**, seleccione **Default** y haga click **Edit**.



c.- Ingrese *Development* y presione **OK**.



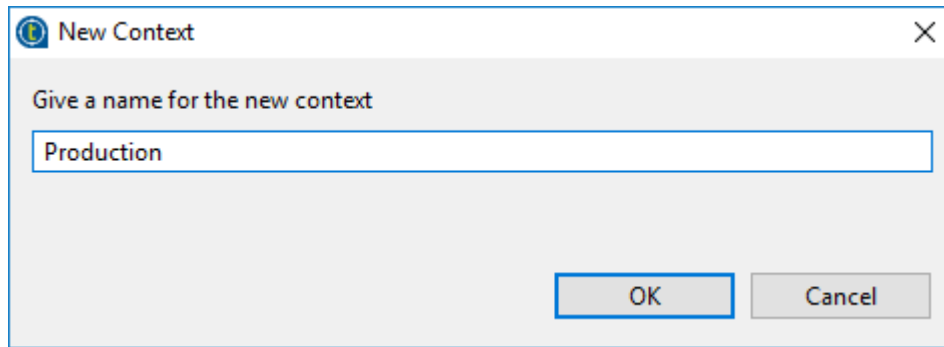
El contexto aparece con un nuevo nombre.



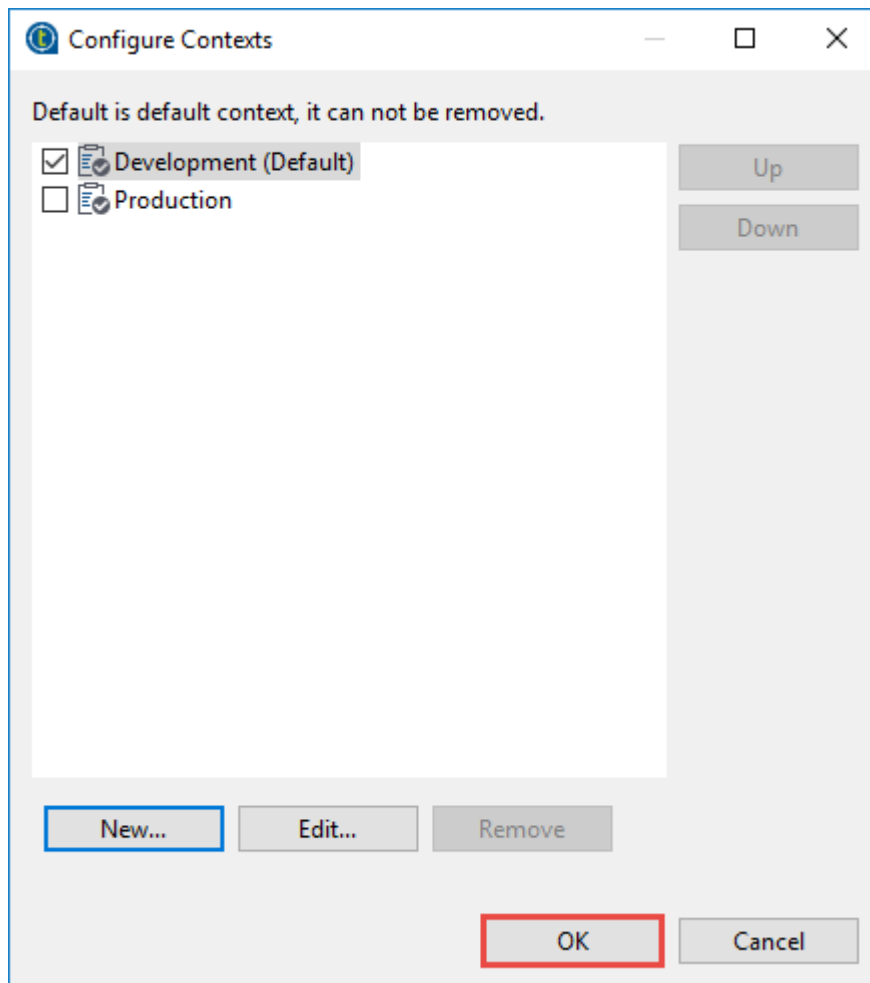
2.- En la misma ventana, cree el contexto *Production*.

a.- En la ventana **Configure Contexts**, click **New**.

b.- Ingrese *Production* y presione **OK**.



c.- Para cerrar la ventana, click **OK**.

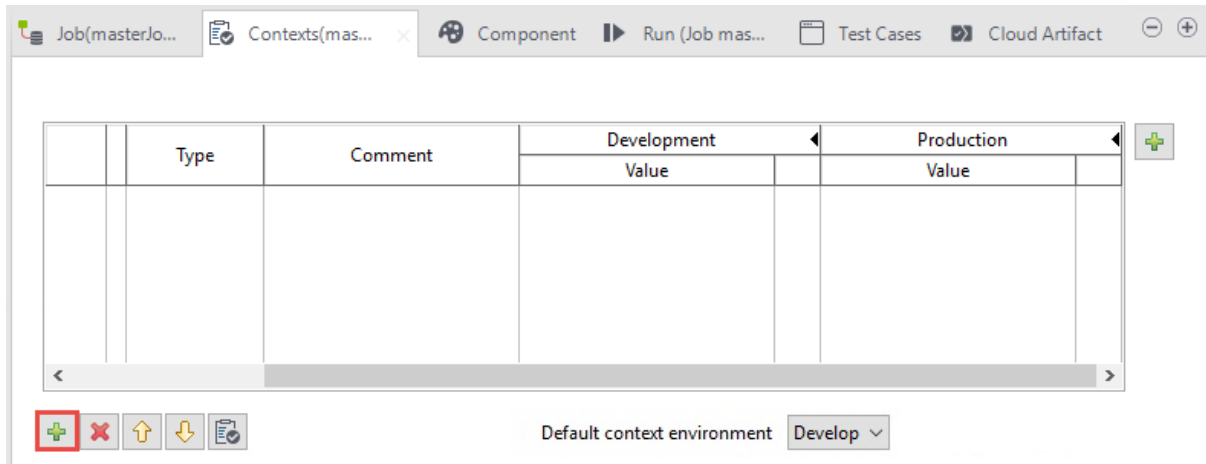


En la pestaña Contexts, el contexto *Production* aparecerá en una nueva columna.

3.- Cree una variable de contexto built-in.

Cree la variable *StagingDirectory*, y para ambos contextos, ingrese el valor *C:/StudentFiles/DIBasics/ExecutionControl/MasterContext*.

a.- Para agregar una variable de contexto, en la esquina inferior izquierda de la tabla, haga clic en [+].



b.- Ingrese estos parámetros:

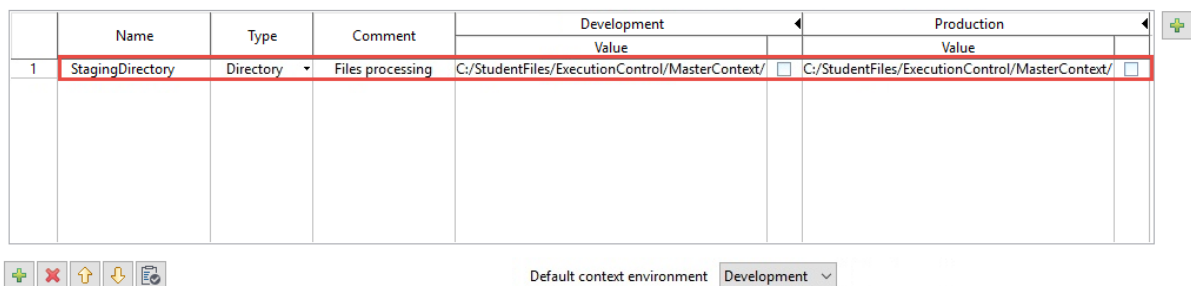
→ Para **Name**, Ingrese *StagingDirectory*.

→ Para **Type**, Ingrese **Directory**.

→ En **Comment**, agregue una descripción.

→ En la columna **Development**, a la derecha de **Value**, click en el botón **square**, busque la ruta **C:/StudentFiles/DIBasics/ExecutionControl**, cree la carpeta **MasterContext** y selecciónela.

→ Copie el **Value** de development y péguelo en la columna de **Production**.



La variable está lista para ser usada

💡 También puede ingresar la ruta manualmente.

4.- Use el contexto master Job.

Para transmitir la variable StagingDirectory con el valor seleccionado del contexto del master Job, seleccione **Transmit whole context** en ambos componentes **tRunJob**.

a.- En **Designer**, seleccione **filesStaging** y haga click en la pestaña **Component**.

b.- Seleccione Transmit whole context.

filesStaging(tRunJob_1)

Basic settings

Schema: Built-In (v) Edit schema (...) Copy Child Job Schema

Advanced settings

Dynamic settings

Job: filesStaging (...) Version: Latest (v)* Context: Development (v)*

View

Documentation

Validation Rules

☐ Use dynamic job

☐ Use an independent process to run subjob

☒ Die on child error

☒ Transmit whole context

Context Param	Parameters	Values
---------------	------------	--------

c.- Haga lo mismo para **processDirectory**.

5.- Ejecute el master Job y observe el mensaje que muestra en pantalla.

a.- Abra la pestaña Run.

b.- Click Run. El Job debe finalizar con **[exit code=0]**.

[illegible]

El mensaje que se muestra en la consola muestra que la ruta utilizada fue la que seleccionó del contexto del master Job.

c.- Navegue a *C:\Backup*.

Confirme que se ha generado un nuevo archivo.

6.- Configure el valor de la variable.

Para transmitir una variable `StagingDirectory` de forma manual, cree una nueva entrada en **Context Param** para ambos componentes **tRunJob**.

a.- En **Designer**, seleccione **fileStaging** y haga click la pestaña **Component**.

b.- Para añadir una variable de contexto, en la esquina inferior izquierda de la tabla, click [+].

The screenshot shows the configuration for the **fileStaging(tRunJob_1)** component. The left sidebar contains tabs for **Basic settings**, **Advanced settings**, **Dynamic settings**, **View**, **Documentation**, and **Validation Rules**. The main area displays the **Context Param** table, which is currently empty. Below the table, a row of icons is visible, with the first icon (a green plus sign) highlighted by a red rectangle.

c.- Utilice los siguientes parámetros:

→ Para **Parameter**, seleccione **StagingDirectory**

→ Para **Value**, ingrese

"C:/StudentFiles/DIBasics/ExecutionControl/ContextParam/"

💡 Asegúrese de encerrar el valor entre comillas dobles.

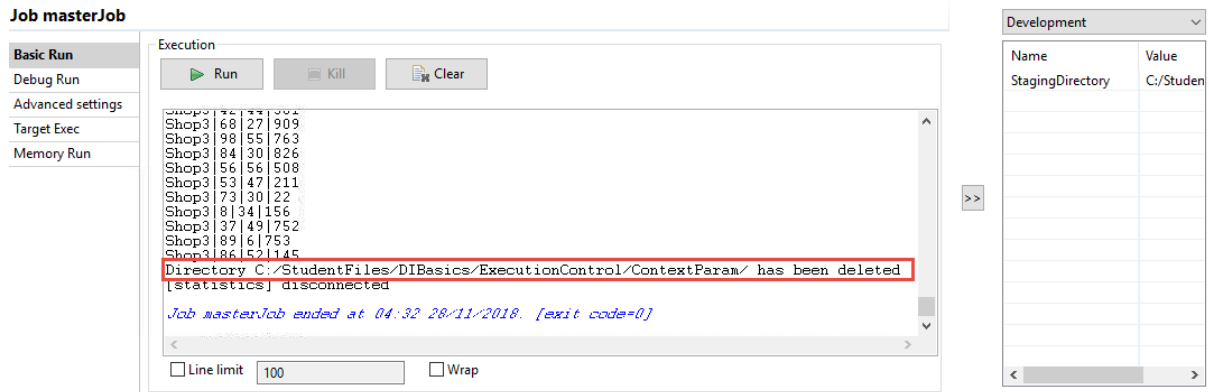
This screenshot shows the same configuration as the previous one, but now the **Context Param** table contains one entry. The entry has the parameter **StagingDirectory** and the value **"C:/StudentFiles/DIBasics/ExecutionControl/ContextParam/"**. The entire row in the table is highlighted with a red border.

d.- Repita el proceso para **processDirectory**.

7.- Ejecute el master Job y observe el mensaje de la consola.

a.- Abra la pestaña Run.

b.- Click Run. El Job debe finalizar con **[exit code=0]**.



El mensaje en la consola muestra que la ruta utilizada fue la que se seleccionó para los componentes de tRunJob.

c.- Navegue a *C:\Backup*.

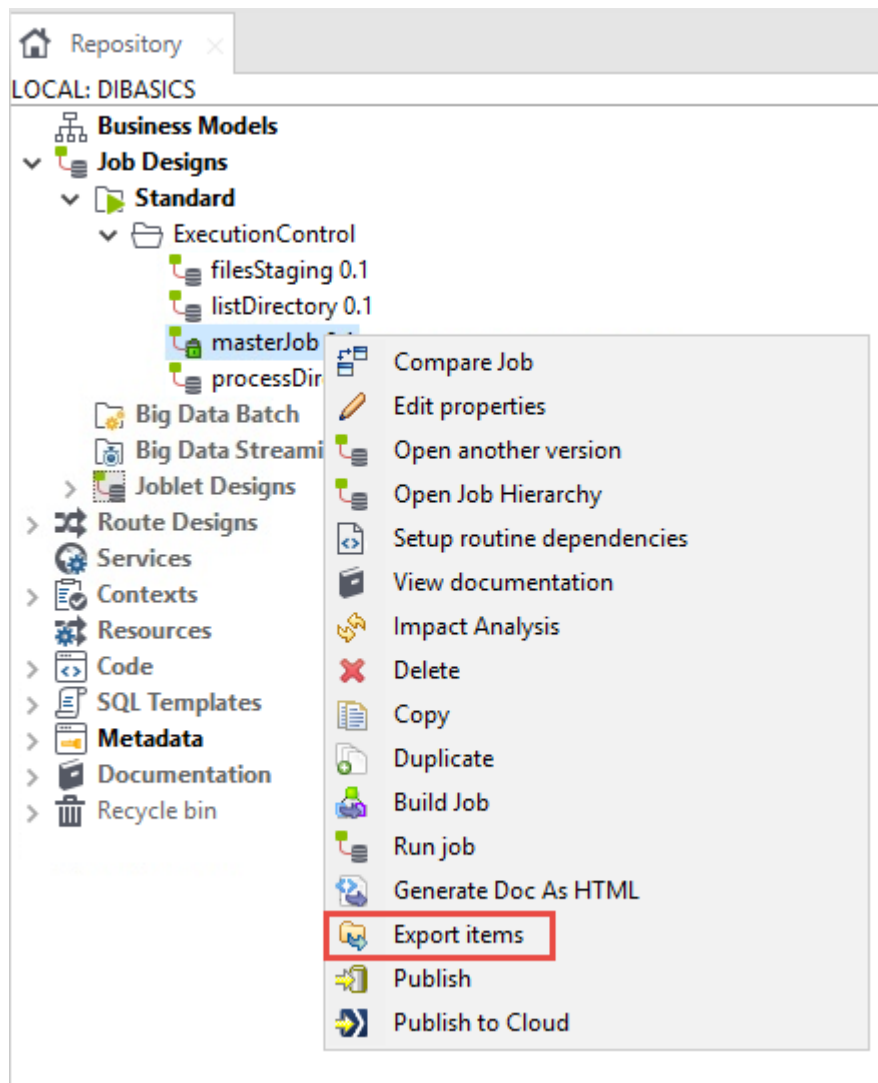
Confirme que se ha generado un nuevo archivo.

Exportar un master Job

Ahora que el master Job se ha probado con éxito, el siguiente paso es exportarlo para que pueda enviarlo a otra persona para su uso futuro.

1.- Exporte **masterJob** como un archivo de almacenamiento con todas las dependencias (todo lo que se necesita para ejecutar el Job se debe incluir en el archivo de almacenamiento).

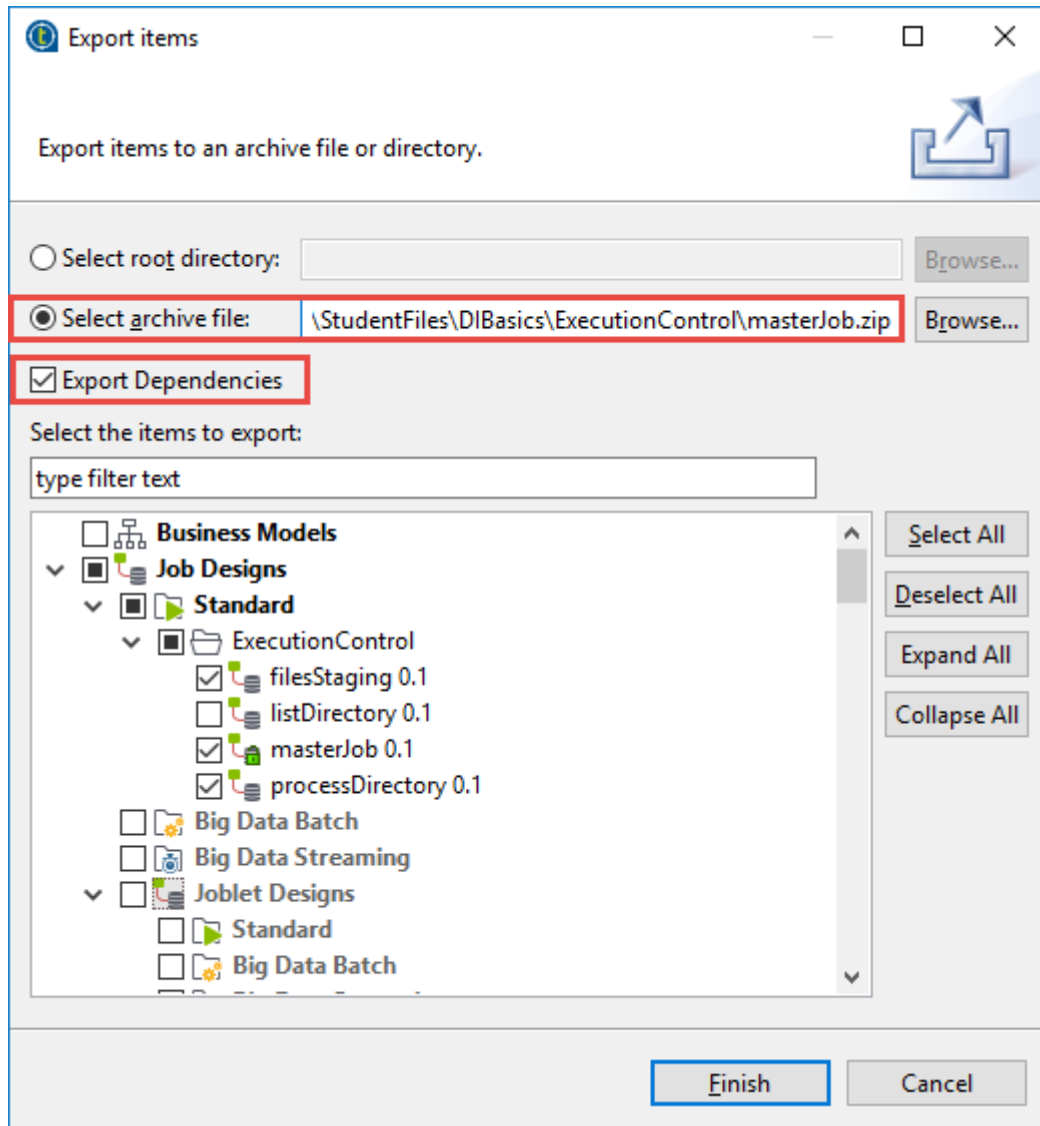
a.- En **Repository**, expanda **Job Designs > Standard > ExecutionControl**, haga click derecho sobre **masterJob** y seleccione **Export items**.



b.- En la ventana **Export items**, seleccione **Select archive file** e ingrese *C:/StudentFiles/DIBasics/ExecutionControl/masterJob.zip*.

💡 También puede utilizar el botón Browse para buscar la ruta.

c.- Seleccione **Export Dependencies**.



Observe que al habilitar la opción Export Dependencies se selecciona automáticamente componentes adicionales para exportar.

d.- Click **Finish**.

2.- Verifique el archivo de almacenamiento.

Navegue a *C:/StudentFiles/DIBasics/ExecutionControl* y confirme que el archivo **MasterJob.zip** ha sido creado ahí.

Resumen

En este módulo aprendiste cómo:

- Usar componentes para listar, archivar y eliminar archivos de un directorio.
- Administrar iteraciones dentro de un Job.
- Reutilizar variables de componente en otros componentes.
- Usar triggers para conectar componentes y subJobs.
- Crear un master Job.
- Transmitir variables de contexto a subJobs.
- Exportar un master Job y sus dependencias.