## Contents

**Copyleft**

**Introduction to Talend Open Studio for Big Data**

Functional architecture of Talend Open Studio for Big Data

**Prerequisites to using Talend Open Studio for Big Data**

- Memory requirements
- Software requirements
- Installing Java
- Setting up the Java environment variable on Windows
- Setting up the Java environment variable on Linux
- Installing 7-Zip (Windows)

**Downloading and installing Talend Open Studio for Big Data**

- Downloading Talend Open Studio for Big Data
- Installing Talend Studio

**Configuring and setting up your Talend product**

- Launching the Studio for the first time
- Logging on to the Studio
- Installing additional packages
- Uploading files to DBFS (Databricks File System)
- Preparing the movies metadata

**Performing data integration tasks for Big Data**

- Joining movie and director information
- What’s next?
XmlBeans, XmlSchema Core,.Xmlsec - Apache Santuario, YAML parser and emitter for Java, Zip4J, atinject, dropbox-sdk-java: Java library for the Dropbox Core API, google-guice. Licensed under their respective license.
Introduction to Talend Open Studio for Big Data

Talend provides unified development and management tools to integrate and process all of your data with an easy to use, visual designer.

Built on top of Talend’s data integration solution, the big data solution is a powerful tool that enables users to access, transform, move and synchronize big data by leveraging the Apache Hadoop Big Data Platform and makes the Hadoop platform ever so easy to use.

Functional architecture of Talend Open Studio for Big Data

The Talend Open Studio for Big Data functional architecture is an architectural model that identifies Talend Open Studio for Big Data functions, interactions and corresponding IT needs. The overall architecture has been described by isolating specific functionalities in functional blocks.

The following chart illustrates the main architectural functional blocks.

The different types of functional blocks are:

- From Talend Studio, you design and launch Big Data Jobs that leverage a Hadoop cluster to handle large data sets. Once launched, these Jobs are sent to, deployed on and executed on this Hadoop cluster.
- The subscription-based workflow scheduler system allows you to deploy, schedule, and execute Big Data Jobs on a Hadoop cluster and monitor the execution status and results of these Jobs.
- A Hadoop cluster independent of the Talend system to handle large data sets.
Prerequisites to using Talend Open Studio for Big Data

This chapter provides basic software and hardware information required and recommended to get started with your Talend Open Studio for Big Data.

• Memory requirements on page 6
• Software requirements on page 6

It also guides you to install and configure required and recommended third-party tools:

• Installing Java on page 7
• Setting up the Java environment variable on Windows on page 7 or Setting up the Java environment variable on Linux on page 8
• Installing 7-Zip (Windows) on page 8

Memory requirements

To make the most out of your Talend product, please consider the following memory and disk space usage:

<table>
<thead>
<tr>
<th>Memory usage</th>
<th>3GB minimum, 4 GB recommended</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disk space</td>
<td>3GB</td>
</tr>
</tbody>
</table>

Software requirements

To make the most out of your Talend product, please consider the following system and software requirements:

Required software

• Operating System for Talend Studio:

<table>
<thead>
<tr>
<th>Support type</th>
<th>Operating system (64 bits only)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recommended</td>
<td>Ubuntu 18.04 LTS</td>
</tr>
<tr>
<td>Recommended</td>
<td>Microsoft Windows 10</td>
</tr>
<tr>
<td>Supported</td>
<td>Apple macOS 10.14/Mojave</td>
</tr>
<tr>
<td></td>
<td>Apple macOS 10.13/High Sierra</td>
</tr>
<tr>
<td></td>
<td>Apple macOS 10.12/Sierra</td>
</tr>
</tbody>
</table>

• You should install either Oracle Java 8 or 11 JRE, or OpenJDK 1.8 or 11.
• A properly installed and configured Hadoop cluster.

Ensure that the client machine on which the Talend Studio is installed can recognize the host names of the nodes of the Hadoop cluster to be used. For this purpose, add the IP address/
hostname mapping entries for the services of that Hadoop cluster in the `hosts` file of the client machine.

For example, if the host name of the Hadoop Namenode server is `talend-cdh550.weave.local`, and its IP address is `192.168.x.x`, the mapping entry reads `192.168.x.x talend-cdh550.weave.local`.

Optional software
- 7-Zip.

Installing Java

To use your Talend product, you need a JAVA environment installed on your computer.

This documentation is about Oracle JRE installation. For more information on OpenJDK, see [http://openjdk.java.net/](http://openjdk.java.net/).

Procedure
1. From the Java SE Downloads page, under *Java Platform, Standard Edition*, click the JRE Download.
2. From the *Java SE Runtime Environment 8 Downloads* page, click the radio button to Accept License Agreement.
3. Select the appropriate download for your Operating System.
4. Follow the Oracle installation steps to install Java.

Results
When Java is installed on your computer, you need to set up the `JAVA_HOME` environment variable. For more information, see:
- Setting up the Java environment variable on Windows on page 7.
- Setting up the Java environment variable on Linux on page 8.

Setting up the Java environment variable on Windows

Prior to installing your Talend product, you need to set the `JAVA_HOME` and Path environment variables.

Procedure
1. Go to the Start Menu of your computer, right-click on *Computer* and select Properties.
2. In the Control Panel Home window, click Advanced system settings.
3. In the System Properties window, click Environment Variables....
4. Under System Variables, click New... to create a variable. Name the variable `JAVA_HOME`, enter the path to the Java 8 JRE, and click OK.

   Example of default JRE path: `C:\Program Files\Java\jre1.8.0_77`.

5. Under System Variables, select the Path variable and click Edit... to add the previously defined `JAVA_HOME` variable at the end of the Path environment variable, separated with semi colon.

   Example: `<PathVariable>;%JAVA_HOME%\bin`. 

7
Prerequisites to using Talend Open Studio for Big Data

Setting up the Java environment variable on Linux

Prior to installing your Talend product, you have to set the JAVA_HOME and Path environment variables.

Procedure

1. Find the JRE installation home directory.
   
   Example: `/usr/lib/jvm/jre1.8.0_65`

2. Export it in the JAVA_HOME environment variable.
   
   Example:
   ```
   export JAVA_HOME=/usr/lib/jvm/jre1.8.0_65
   export PATH=$JAVA_HOME/bin:$PATH
   ```

3. Add these lines at the end of the user profiles in the `~/.profile` file or, as a superuser, at the end of the global profiles in the `/etc/profile` file.

4. Log on again.

Installing 7-Zip (Windows)

Talend recommends to install 7-Zip and to use it to extract the installation files: [http://www.7-zip.org/download.html](http://www.7-zip.org/download.html).

Procedure

1. Download the 7-Zip installer corresponding to your Operating System.

2. Navigate to your local folder, locate and double-click the 7z exe file to install it.

Results

The download will start automatically.
Downloading and installing Talend Open Studio for Big Data

Talend Open Studio for Big Data is easy to install. After downloading it from Talend’s Website, a simple unzipping will install it on your computer.

This chapter provides basic information useful to download and install it.

Downloading Talend Open Studio for Big Data

Talend Open Studio for Big Data is a free open source product that you can download directly from Talend’s Website.

Procedure

2. When prompted, click Save File and then OK.

Results

The zip file of download Talend Open Studio for Big Data is downloaded.

Installing Talend Studio

Unzip the archive previously downloaded to install Talend Studio.

You can do it either by using:


Extracting via 7-Zip (Windows recommended)

For Windows, Talend recommends you to install 7-Zip and use it to extract files.

To install the Studio, follow the steps below:

Procedure

1. Navigate to your local folder, locate the ZIP file previously downloaded and move it to another location with a path as short as possible and without any space character.
   
   Example: C:/Talend/

2. Unzip it by right-clicking on the compressed file and selecting 7-Zip > Extract Here.
Extracting via Windows default unzipping tool

If you do not want to use 7-Zip, you can use Windows default unzipping tool.

Procedure
1. Unzip it by right-click the compressed file and select Extract All.
2. Click Browse and navigate to the C: drive.
3. Select Make new folder and name the folder Talend. Click OK.
4. Click Extract to begin the installation.

Extracting via the Linux GUI unzipper

To install the Studio, follow the steps below:

Procedure
1. Navigate to your local folder, locate the previously downloaded ZIP file and move it to another location with a path as short as possible and without any space character.
   
   Example: home/user/talend/
2. Unzip it by right-clicking on the compressed file and selecting Extract Here.
Configuring and setting up your Talend product

This chapter provides basic information required to configure and set up your Talend Open Studio for Big Data.

Launching the Studio for the first time

The Studio installation directory contains binaries for several platforms including Mac OS X and Linux/Unix.

To open the Talend Studio for the first time, do the following:

Procedure

1. Double-click the executable file corresponding to your operating system, for example:
   - TOS_*-win-x86_64.exe, for Windows.
   - TOS_*-linux-gtk-x86_64, for Linux.
   - TOS_*-macosx-cocoa.app, for Mac.
2. In the User License Agreement dialog box that opens, read and accept the terms of the end user license agreement to proceed.

Logging on to the Studio

To log on to the Talend Studio for the first time, do the following:

Procedure

1. In the Talend Studio login window, select Create a new project, specify the project name: getting_started and click Finish to create a new local project.
2. Depending on the product you are using, either of the following opens:
   - the Quick Tour. Play it to get more information on the User Interface of the Studio, and click Stop to end it.
   - the Welcome page. Follow the links to get more information about the Studio, and click Start Now! to close the page and continue opening the Studio.

Tip:
After your Studio successfully launches, you can also click the Videos link on the top of the Studio main window to watch a couple of short videos that help you get started with your Talend Studio. For some operating systems, you may need to install an MP4 decoder/player to play the videos.

Results

Now you have successfully logged on to the Talend Studio. Next you need to install additional packages required for the Talend Studio to work properly.
Installing additional packages

Talend recommends that you install additional packages, including third-party libraries and database drivers, as soon as you log in to your Talend Studio to allow you to fully benefit from the functionalities of the Studio.

Before you begin

**Warning:** Make sure that the -Dtalend.disable.internet parameter is not present in the Studio.ini file or is set to false.

Procedure

1. When the Additional Talend Packages wizard opens, install additional packages by selecting the **Required** and **Optional third-party libraries** check boxes and clicking **Finish**.

   This wizard opens each time you launch the studio if any additional package is available for installation unless you select the **Do not show this again** check box. You can also display this wizard by selecting Help > Install Additional Packages from the menu bar.

   For more information, see the section about installing additional packages in the Talend Open Studio for Big Data Installation and Upgrade Guide

2. In the **Download external modules** window, click the **Accept all** button at the bottom of the wizard to accept all the licenses of the external modules used in the studio.

   Depending on the libraries you selected, you may need to accept their license more than once.

   Wait until all the libraries are installed before starting to use the studio.

3. If required, restart your Talend Studio for certain additional packages to take effect.

Uploading files to DBFS (Databricks File System)

Uploading a file to DBFS allows the Big Data Jobs to read and process it. DBFS is the Big Data file system to be used in this example.

In this procedure, you will create a Job that writes data in your DBFS system. For the files needed for the use case, download tos_bd_gettingstarted_source_files.zip from the Downloads tab of the online version of this page at https://help.talend.com.

Procedure

1. In the **Repository** tree view, right click the **Job Designs** node, and select Create folder from the contextual menu.

2. In the **New Folder** wizard, name your Job folder **getting_started** and click **Finish** to create your folder.

3. Right-click the **getting_started** folder and select Create Job from the contextual menu.

4. In the **New Job** wizard, give a name to the Job you are going to create and provide other useful information if needed.

   For example, enter write_to_dbfs in the **Name** field.
In this step of the wizard, **Name** is the only mandatory field. The information you provide in the **Description** field will appear as hover text when you move your mouse pointer over the Job in the **Repository** tree view.

5. Click **Finish** to create your Job.

An empty Job is opened in the Studio.

6. In the design space of this empty Job, type `dbfs` to search for the DBFS related components. On the component list that is displayed, double-click **tDBFSConnection** to select it. The **tDBFSConnection** is added to the design space.

7. Repeat this operation to add tDBFSPut to the design space.

8. Right click **tDBFSConnection** and from the contextual menu that is displayed, select **Trigger > On Subjob Ok**.

**Example**

9. Click **tDBFSPut** to connect **tDBFSConnection** to **tDBFSPut**.

10. Double-click **tDBFSConnection** to open its **Component** view.

**Example**

11. In the **Endpoint** field, enter the URL address of your Azure Databricks workspace. This URL can be found in the **Overview** blade of your Databricks workspace page on your Azure portal. For example, this URL could look like https://westeurope.azuredatabricks.net.

12. Click the `[...]` button next to the **Token** field to enter the authentication token generated for your Databricks user account. You can generate or find this token on the **User settings** page of your Databricks workspace. For further information, see **Token management** from the Azure documentation.
13. Double-click tDBFSPut to open its Component view.

**Example**

![Component view of tDBFSPut](image)

14. Select **Use an existing connection** to use the connection information defined in tDBFSConnection.
15. In the **Local directory** field, enter the path, or browse to the folder in which the files to be copied to DBFS are stored.
16. In the **DBFS directory** field, enter the path to the target directory in DBFS to store the files. This location is recommended to be in the FileStore folder, according to the FileStore section in the Databricks documentation.
   
   This directory is created on the fly if it does not exist.
17. From the **Overwrite file** drop-down list, select **always** to overwrite the files if they already exist in the target directory in DBFS.
18. In the **Files** table, add one row by clicking the [+] button in order to define the criteria to select the files to be copied.
19. In the **Filemask** column, enter an asterisk (*) within the double quotation marks to make tDBFSPut select all the files stored in the folder you specified in the **Local directory** field.
20. Leave the **New name** column empty, that is to say, keep the default double quotation marks as is, so as to make the name of the files unchanged after being uploaded.
21. Press F6 to run the Job.

The RunThe files about movies and their directors are stored in this view is opened automatically. It shows the progress of this Job.
Results

When the Job is done, the files you uploaded can be found in DBFS in the directory you have specified.

Preparing the movies metadata

This example describes how to set up the metadata of the source file movies.csv in the Repository. Repository metadata can be used across Jobs, allowing you to configure your Jobs quickly without having to define each parameter and schema manually.

Before you begin

- You have the source file movies.csv ready in the directory C:\getting_started\input_data\.
Procedure

1. In the Repository tree view, expand the Metadata node, right-click File delimited, and select Create file delimited from the contextual menu to open the New Delimited File wizard.

2. In the New Delimited File wizard, enter a name for the file metadata, movies in this example, and other useful information to better describe your file metadata, and then click Next to go to the next step and define the general properties of the file.

   ![New Delimited File Wizard](image)

   In this step of the wizard, Name is the only mandatory field. The information you provide in the Description field will appear as a tooltip when you move your mouse pointer over the file connection.

3. In the File field specify the path of the source file, or click Browse to browse to the file.
The file is loaded, and the File Viewer area displays an abstract of the file, allowing you to check the file consistency, the presence of header and more generally the file structure.

4. From the Format list, select your operating system, and click Next to parse the file.

5. On the Preview tab, select the Set heading row as column names check box to retrieve the file column names from the first row, and then click Refresh Preview.
Configuring and setting up your Talend product

The file preview is refreshed, and the **Header** check box in the **Rows To Skip** area is automatically selected, with the number of header rows to be skipped incremented by 1.

6. If the file contains more than one heading row, which need to be skipped in file parsing, specify the number in this field and click **Refresh Preview** again.

7. Click **Next** to retrieve the file schema.

   The **Description of the Schema** table displays the generated file schema.

8. Name the schema **movies_schema** and check the file schema and edit it according to your actual needs.

   In this example, increase the length of the **title** and **url** columns.
9. Click **Finish** to validate the schema close the wizard.

The created file metadata is shown in the **Repository** tree view.

![Repository tree view with movies_schema](image)

**Results**

You now have the movies file metadata ready for use. Next, you need to apply the created metadata to the component that reads the source file.
Performing data integration tasks for Big Data

This chapter takes the example of a company that provides movie rental and streaming video services, and shows how such a company could make use of Talend Open Studio for Big Data.

You will work with data about movies and directors and data about your customers as you learn how to:

- load data in the data flow of your Job from the file system of the company’s Big Data platform. In this example, this platform is Azure Databricks and this file system is DBFS.
- join the director data to the movie data to produce a new dataset and store this dataset in an Azure Blob Storage container.

Joining movie and director information

This scenario demonstrates:

1. How to create a Talend Job. See Creating the Job on page 20 for details.
2. How to drop and link the components to be used in a Job. See Dropping and linking components on page 21 for details.
3. How to configure the input components using the related metadata from the Repository. See Configuring how to read the input data on page 22 for details.
4. How to configure the transformation to join the input data. See Configuring the data transformation component on page 25 for details.
5. How to write the transformed data to an Azure Blob Storage container. See Writing the output on page 27 for details.

Creating the Job

A Talend Job allows you to access and use the Talend components to design technical processes to read, transform or write data.

Before you begin

- You have launched your Talend Studio and opened the Integration perspective.

Procedure

1. Right-click the getting_started folder and select Create Job from the contextual menu.
2. In the New Job wizard, give a name to the Job you are going to create and provide other useful information if needed.

   For example, enter aggregate_movie_director in the Name field.

   In this step of the wizard, Name is the only mandatory field. The information you provide in the Description field will appear as hover text when you move your mouse pointer over the Job in the Repository tree view.

3. Click Finish to create your Job.

   An empty Job is opened in the Studio.
Results

The component **Palette** is now available in the Studio. You can start to design the Job by leveraging this **Palette** and the **Metadata** node in the **Repository**.

**Dropping and linking components**

The DBFS, Azure and processing components to be used are orchestrated in the Job workspace to compose a complete process for data transformation.

**Before you begin**

- You have launched your Talend Studio and opened the **Integration** perspective.
- An empty Job has been created as described in **Creating the Job** on page 20 and is open in the workspace.

**Procedure**

1. In the Job, enter the name of the component to be used and select this component from the list that appears. In this scenario, the components are two **tFileInputDelimited** components, a **tMap** component, two **tFileOutputDelimited** components, a **tDBFSConnection** component, a **tDBFSGet** component and a **tAzureStoragePut**.
   - The DBFS components connect to your Databricks file system (DBFS) to download the files about movies and directors.
   - The two **tFileInputDelimited** components are used to load the movie data and the director data, respectively, from your local file system into the data flow of the current Job.
   - The **tMap** component is used to transform the input data.
   - The **tFileOutputDelimited** components write the results into given directories in your local system.
   - The **tAzureStoragePut** component is used to upload the transformed data in an Azure Blob Storage container.
2. Double-click the label of one of the **tFileInputDelimited** component to make this label editable and then enter **movie** to change the label of this component.
3. Do the same to label the other **tFileInputDelimited** component to **director**.
4. Right click **tDBFSConnection** and from the contextual menu that is displayed, select **Trigger > On Subjob Ok**.
5. Click **tDBFSGet** to connect **tDBFSConnection** to **tDBFSGet**.
6. Repeat the same operations to always use the **On Subjob Ok** link to connect **tDBFSGet** to the **tFileInputDelimited** component labelled **movie**, then connect the same **tFileInputDelimited** component to **tAzureStoragePut**.
7. Right click the **tFileInputDelimited** component that is labelled **movie**, then from the contextual menu, select **Row > Main** and click **tMap** to connect these two components. This is the main link through which the movie data is sent to **tMap**.
8. Do the same to connect the director **tFileInputDelimited** component to **tMap** using the **Row > Main** link. This is the Lookup link through which the director data is sent to **tMap** as lookup data.
9. Do the same to connect the **tMap** component to **tFileOutputDelimited** using the **Row > Main** link, then in the pop-up wizard, name this link to **out1** and click **OK** to validate this change.
10. Repeat these operations to connect the **tMap** component to the other **tFileOutputDelimited** component using the **Row > Main** link and name it to **reject**.
Performing data integration tasks for Big Data

Results

Now the whole Job looks as follows in the workspace:

Configuring how to read the input data

The DBFS components and the two tFileInputDelimited components are configured to load data from DBFS into the Job.

Before you begin

- The source files, movies.csv and directors.txt have been uploaded to DBFS as explained in Uploading files to DBFS (Databricks File System) on page 12.
- The metadata of the movie.csv file has been set up under the File delimited node in the Repository.
  
  If you have not done so, see Preparing the movies metadata on page 15 to create the metadata.

Procedure

1. Double-click tDBFSConnection to open its Component view.
Example

2. In the **Endpoint** field, enter the URL address of your Azure Databricks workspace. This URL can be found in the **Overview** blade of your Databricks workspace page on your Azure portal. For example, this URL could look like https://westeurope.azuredatabricks.net.

3. Click the [...] button next to the **Token** field to enter the authentication token generated for your Databricks user account. You can generate or find this token on the **User settings** page of your Databricks workspace. For further information, see Token management from the Azure documentation.

4. Double-click tDBFSGet to open its **Component** view.

Example

5. Select **Use an existing connection** to use the connection information defined in tDBFSCConnection.

6. In the **DBFS directory** field, enter the path to the directory in DBFS in which the files about movies and their directors are stored.

7. In the **Local directory** field, enter the path, or browse to the folder in which the files to be downloaded from DBFS are stored.

   This directory is created on the fly if it does not exist.

8. From the **Overwrite file** drop-down list, select **always** to overwrite the files if they already exist in the target directory in the local file system.

9. In the **Files** table, add one row by clicking the [+] button in order to define the criteria to select the files to be copied.

10. In the **Filemask** column, enter an asterisk (*) within the double quotation marks to make tDBFSGet select all the files stored in the folder you specified in the **Local directory** field.
11. Leave the New name column empty, that is to say, keep the default double quotation marks as is, so as to make the name of the files unchanged after being uploaded.

12. Expand the File delimited node under the Metadata node in the Repository to display the movies schema metadata node you have set up as explained in Preparing the movies metadata on page 15.

13. Drop this schema metadata node onto the movie tFileInputDelimited component in the workspace of the Job.

14. Double-click the movie tFileInputDelimited component to open its Component view.

This tFileInputDelimited has automatically reused the movie metadata from the Repository to define the related parameters in its Basic settings view.

15. Click the File name/Stream field to open the Edit parameter using repository dialog box to update the field separator.

This tFileInputDelimited is reusing the default file location which you have defined for the File delimited metadata. You need to change it to read the movie file from theh directory in which this file is downloaded from DBFS.

16. Select Change to built-in property and click OK to validate your choice.

The File name/Stream field becomes editable.

17. Enter the directory where the movie file downloaded from DBFS is stored.

18. Double-click the director tFileInputDelimited component to open its Component view.
19. Click the [...] button next to **Edit schema** to open the schema editor.

20. Click the [+ ] button twice to add two rows and in the **Column** column, rename them to **ID** and **Name**, respectively.

21. Click **OK** to validate these changes and accept the propagation prompted by the pop-up dialog box.

22. In the **File name/Stream** field, enter the directory where the data about the movie directors is stored.

23. In the **Field separator** field, enter a comma (,) within double quotation marks.

**Results**

The **tFileInputDelimited** components are now configured to load the movie data and the director data to the Job.

**Configuring the data transformation component**

The **tMap** component is configured to join the movie data and the director data.

Once the movie data and the director data are loaded into the Job, you need to configure the **tMap** component to join them to produce the output you expect.
Performing data integration tasks for Big Data

Procedure

1. Double-click tMap to open its Map Editor view.

2. Drop the movieID column, the title column, the releaseYear column and the url column from the left side onto each of the output flow table.

   On the input side (left side) of the Map Editor, each of the two tables represents one of the input flow, the upper one for the main flow and the lower one for the lookup flow.

   On the output side (right side), the two tables represent the output flows that you named to out1 and reject when you linked tMap to tFileOutputDelimited in Dropping and linking components on page 21.

3. On the input side, drop the directorID column from the main flow table to the Expr.key column of the ID row in the lookup flow table.

   This way, the join key between the main flow and the lookup flow is defined.

4. Drop the directorID column from the main flow table to the reject table on the output side and drop the Name column from the lookup flow table to the out1 table.

   The configuration in the previous two steps describes how the columns of the input data are mapped to the columns of the output data flow.

   From the Schema editor view in the lower part of the editor, you can see the schemas on both sides have been automatically completed.

5. On the out1 output flow table, click the button to display the editing field for the filter expression.

6. Enter row1.directorId != null

   This allows tMap to output only the movie records in each of which the directorID field is not empty. A record with an empty directorID field is filtered out.
Performing data integration tasks for Big Data

7. On the reject output flow table, click the button to open the settings panel.
8. In the Catch Output Reject row, select true to output the records with empty directorID fields in the reject flow.
9. Click Apply, then click OK to validate these changes and accept the propagation prompted by the pop-up dialog box.

Results

The transformation is now configured to complete the movie data with the names of their directors and write the movie records that do not contain any director data into a separate data flow.

Writing the output

Two tFileInputDelimited components are configured to write the expected movie data and the rejected movie data to different directories in your local file system; then tAzureStoragePut uploads these files to an existing Azure Blob container.

Procedure

1. Double-click the tFileOutputDelimited which receives the out1 link.
   Its Basic settings view is opened in the lower part of the Studio.

   ![Image of tFileOutputDelimited settings]

   2. In the File Name field, enter the directory you need to write the result in. In this scenario, it is C:/tos_bd_gettingstarted_source_files/temps/out.csv, which receives the records that contain the names of the movie directors.
   3. In the Field separator field, enter ; within double quotation marks.
   4. Click Advanced settings tab to open its view and clear the Throw an error if the file already exists check box to allow the component overwriting existing files of the same names.
   5. Repeat the same operations to configure the tFileOutputDelimited that receives the reject link, but set the directory, in the File Name field, to C:/tos_bd_gettingstarted_source_files/temps/reject.csv.
   6. Double-click the tAzureStoragePut to open its Component view.
7. In the **Account Name** field and the **Account Key** field, enter the credentials of the Azure Blob Storage account to be used. Ensure that the administrator of the system has granted you the appropriate access permissions to this storage account.

8. In the **Container name** field, enter the name of the Azure Blob container to be used. This container must already exist.

9. In the **Local folder** field, enter the directory in which the **tFileOutputDelimited** components write the output files. In this example, the directory is `C:/tos_bd_gettingstarted_source_files/temps`.

10. In the **Azure storage folder** field, enter the name of the folder to be used to store the data to be uploaded to Azure. If it does not exist, this folder is created on the fly in the container you specified above in the **Container name** field.

11. Press F6 to run the Job.

**Results**

The **Run** view is automatically opened in the lower part of the Studio and shows the execution progress of this Job.

Once done, you can check that the output has been written in the Azure Blob Storage container.
Performing data integration tasks for Big Data

What’s next?

You have seen how Talend Studio helps you manage your big data using Talend Jobs. You have learned how to access and move your data to a given Hadoop cluster via Talend Jobs, filter and transform your data, and store the filtered and transformed data in the HDFS system of the Hadoop cluster. Along the way, you have learned how to centralize frequently used Hadoop connections in the Repository and easily reuse these connections in your Jobs.

To learn more about Talend Studio, see:

• Talend Studio User Guide
• Talend components documentation

To ensure that your data is clean, you can try Talend Open Studio for Data Quality and Talend Data Preparation Free Desktop.

To learn more about Talend products and solutions, visit www.talend.com.