Talend User Component tFileInputTextFlat

Purpose

This component extracts field from flat text files with two different methods:

1. Separate fields between delimiters
2. Separate fields at absolute position

The advantages are:

- Configure field position according to the names in the header line (also by regex)
- Read only the fields needed (unlike tFileInputDelimited)
- Tolerant against enclosure chars within the content
- All error messages contains the line number and the field name or index
- After extracting field content you can use regex to do some post processing
- Can load the value extraction configuration from an external file.

Talend-Integration

This component can be found in the palette under File->Input
This component provides an output and a reject flow and several return values.

Parameters

Property	Content
File name	Full path of the file to import.
Encoding	File encoding
Lines to Skip at Start	Number of lines to skip (In case you specify later you have an header line, don't count it here)
Skip empty lines	Detect empty lines and skip them
Schema	The schema. For date and timestamps you should specify here the pattern.
Has Column Header	Check it if your file provides a column header. This enables following options. The header will be skipped in the main output flow.
Use Header line to find position	The position of a delimited field will be set according to the position of the field name in the header line.
Find column position in header by regex	This option enables you to use a regularly expression in the field configuration “Name in Header”.
Load configuration from file	Allows loading the field configuration from a external file.
Configuration file (.importconfig)	The file containing the configuration. See the paragraph below.
Ignore Not Null Constraints	Avoid throwing Exceptions if a value is null but the schema defines it as not null. This works for all columns.
Field Extraction	See explaining below
Field Separator	Char to separate the fields (only for field extracting by method Delimited Fields)
Text Enclosure	Char to enclosure the field content. It is helpful to read fields with line break as content. Don't escape double quotas here! E.g. for a double quoted field set “”” here. It works even if not all fields are putted in quotas.
Split Row before Field	If it is not allowed to have line breaks in field content, check this. This option helps to check the correct file structure. The performance of the line separation can be increased.
Allow Enclosure within Content	If the enclosure char can occurs in the content (not as enclosure) check this to avoid parsing problems. If switched off the file structure can be checked strict.

Locale For Number Format	To parse the number different then the local pattern, specify here the locale. The default is the English format. E.g. in German we have the following pattern: 999.999,99 In this case set the locale to “de”. Quotation required!
Default Date Pattern	For all date schema column without a defined pattern, this pattern here will be applied. If none of the provided date formats are matching, the component performs a self-test typical for English, French and German formats and applying them. Only if this last attempt fails, the component throws and exception.
Die On Error	If true all errors stops the processing. If false all malformed data sets are send to the reject output flow (if added).

Field Extraction

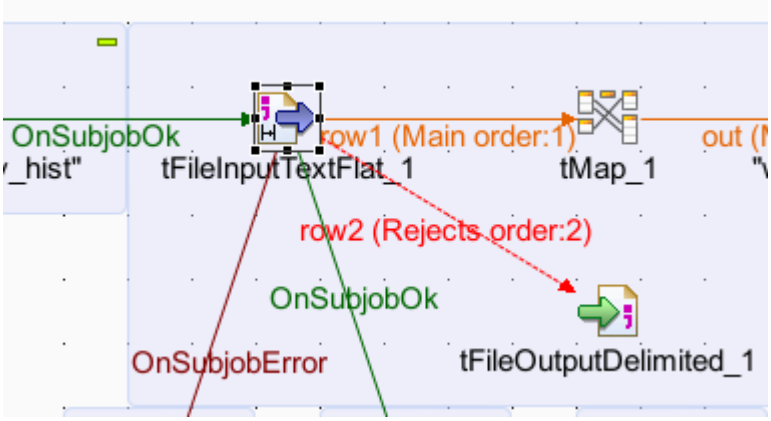
Column	Meaning
Column	Schema column name
Positioning	Field positioning method (extraction method)
Position	Position within delimiters or absolute position (depends on the chosen Positioning method) **If blank for Delimited**: Field position comes from field index in schema **If blank for Absolute Positioning**: Field position is the next after the last (it is a relative position to the previous field)
Length	Length of the field. It is required for absolute positioning. If a length is provided, the content will be trimmed to this length.
Regex	Regularly expression to post processing the field content. (No quotation needed)
Alternative	Set here as String the name of another column which value should be used in case of the own value is empty. This works also cascading -> the alternative column could refer them self to an alternative column. The order of the gathering of the values are: 1. Own default value 2. Alternative column value ….
Name in Header	Specify a different name for the column in the header. Field names often are not given according to the Java identifier naming rules. Therefore, you can specify the real name here (without quotation). If the option “Find column position in header by regex” is switch on you can write here regex (case insensitive). The regex expression must describe the whole possible column name in the header. It is not enough to declare only a pattern like contains. The regex expression will be surrounded from the component with ^ and $. Example: Column name in the header line can vary from: Payment_Product to P_Product. To match a schema column you can write as Name in Header this regex: `P[a-z]*_Product` to match both (and more) possibilities
Ignore If Missing	Sometimes we get fields which does not have all columns expected. You can specify here this column can be missed without problems. If missing the values remains blank.

Return values

Return value	Content
ERROR_MESSAGE	Last error message
NB_LINE	Number of delivered lines
NB_REJECTED	Number of rejected lines

Scenario 1:

Reading a delimited file and write malformed line into a file.



Configuration with examples of different header names.
It is an example of a file, which has as first line a header line.

File Name: context.currentFile

Encoding: UTF-8

Lines to Skip at Start: 0 — Skip Empty Rows

Schema: Built-In — Edit schema

Has Column Header — Use Column Header to Find Position — Ignore Not Null Constraints

Field Extraction

Column	Positioning	Position	Length	Regex	Name in Header	Ignore if missing
Bestelldatum	Delimiter separated					☐
Einkaufspreis	Delimiter separated			"[0-9.]*"		☐
In_Zahlung_gegeben	Delimiter separated		4		In Zahlung gegeben	☐
Gesamtpreis	Delimiter separated					☐
Rabatt	Delimiter separated					☐
Stornierungsdatum	Delimiter separated					☐
Gew__Ankauf	Delimiter separated				Gew. Ankauf	☑

Field Separator: ";"

Text Enclosure: """" — Split Row Before Field — Allow Enclosure within Content

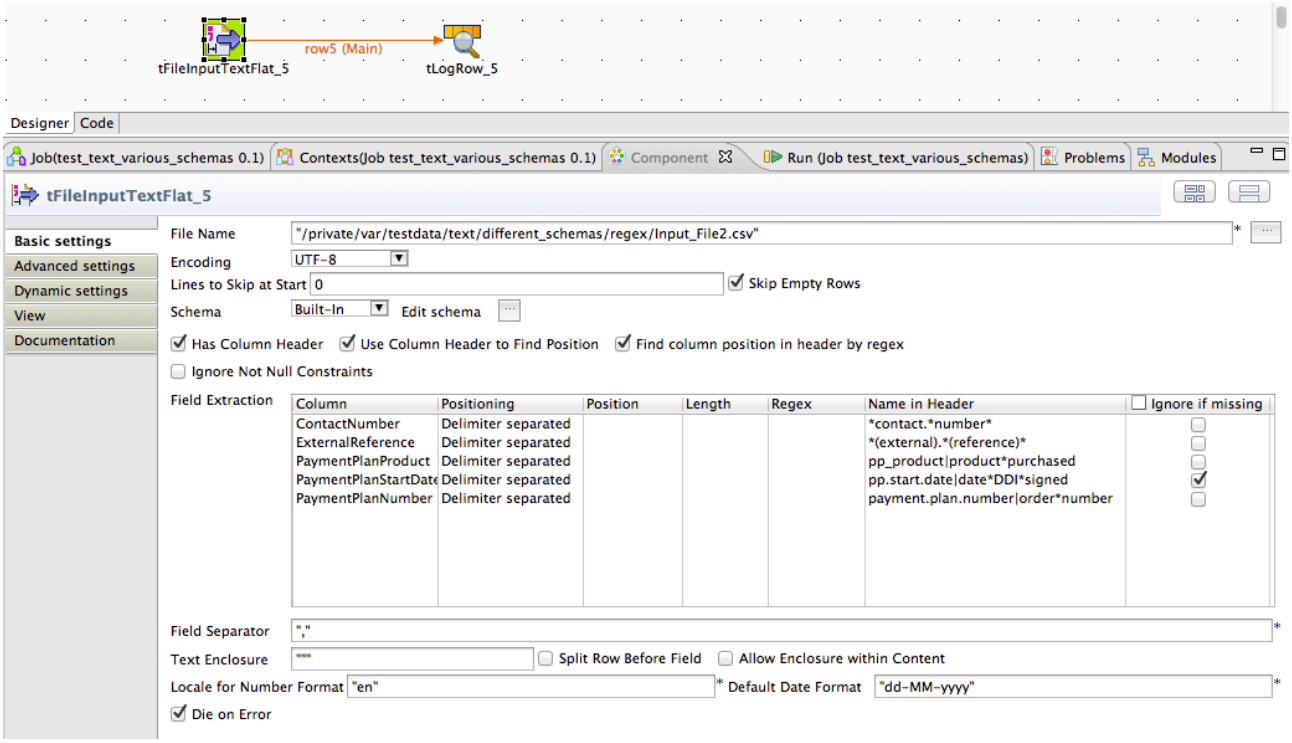
Locale for Number Format: "de" — Default Date Format: "dd.MM.yyy"

Die on Error

It is not necessary to specify the position if the position is identical to the schema column index.

Scenario 2:

Using regularly expression to find the correct field position by the header line of the file.
In case the input files are provided by a system or organization, which cannot be motivated to a more fixed interface design. This should be avoided but sometimes it happens.

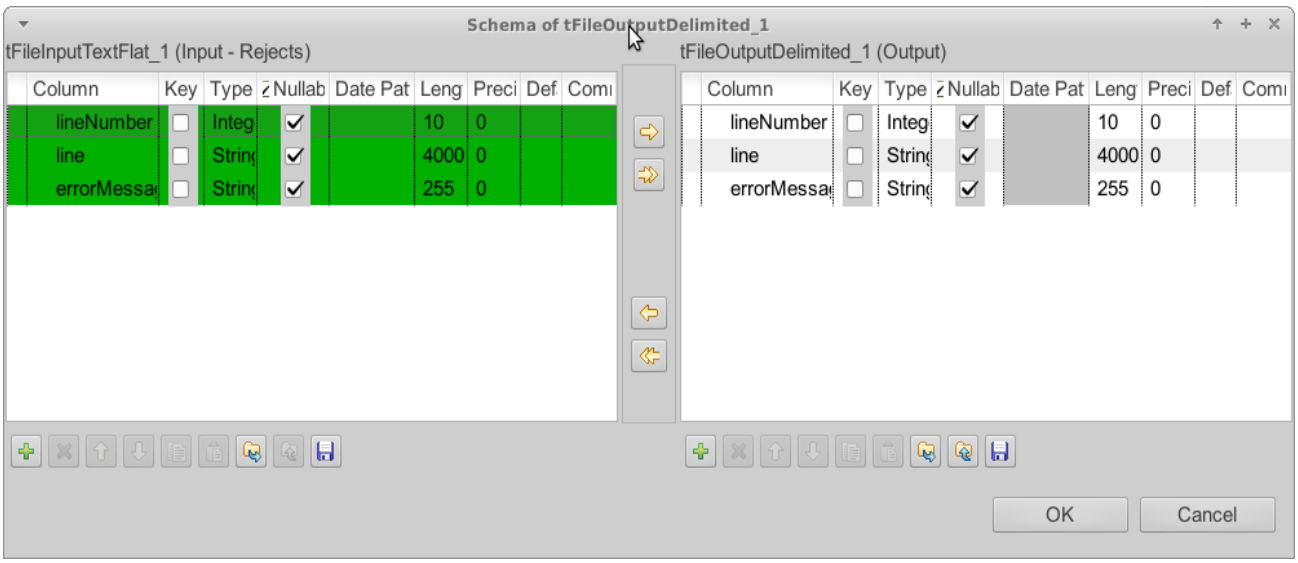


The current example as a header line like this:

```
ContactNumber,ExternalReference,Source,DateDDISigned,Title,ProductPurchased,InterestedInBeingACampaigner,PaymentFrequency,ChangeOfPaymentFrequency,StartDate,NextPaymentDue,AutoPaymentMethodStartDate,MailingDate,CallOutcome,WelcomeCallDate,PaymentMethod,AccountName,PaymentPlanDetailBalance,OrderNumber
```

Only 4 columns are needed and the names can be vary from file to file.
This matching can only be done by regex – see the screenshot above.

Reject flow:
The reject flow will only be filled if the option “Die On Error” is switched off.
If you add the reject flow, at first it contains all columns from the schema and these additional columns. Only the green columns will be filled, all other columns can be erased here.



The line number is the number of the extracted lines and can differ from the number in the file if fields contain line breaks.
The line is the input for the field extraction. You can find here the probably malformed content.
The error message contains the description of the problem occurred while parsing the line.

Possible error messages:

If a column is missing in the header line (assuming we digger for it) an exception will be thrown which says the missing column.

If a column matches to the an already used delimiter position the read of the first line fails with exceptions like this:
`java.lang.Exception: nextDataRow failed in line 0:Extract field pp.start.date|date*DDI*signed failed:Current field index 8 is lower then last field index:9`

This is a failed check of the parser to avoid reading the content with a wrong configuration.
The component sorts all extractions internal by the position. It is a design decision to avoid reading content unwanted twice.

Loading the column configuration from a file

The component has the capability to load the column configuration from a file with the extension .importconfig .
Typical use case is there are many different files which contains the same information but in different columns and positions. Instead of writing different jobs you can write one and describe every file with an import configuration.

The configuration file contains key value pairs. Do not quote (“) the values!
It contains keys to describe the columns and keys to define delimiters and enclosures.

The column index must match to the column index of the schema.
The basic type describes the principle type of the column and can be leaf out as long as the data class in the key CLASS is set.

Key	Type	Description
COLUMN_x_BASICTYPE	Integer	See table below
COLUMN_x_CLASS	String	See table below
COLUMN_x_DEFAULT	String	This is the textual replacement for an empty value. Take as a replacement text in the file.
COLUMN_x_DELIMITERCOUNT	Integer	The position of the field in the row. It starts with 0 and allows gaps.
COLUMN_x_ENABLED	Boolean	Set it to false of you do not want to read this field
COLUMN_x_IGNORE_DATASET_IF_INVALID	Boolean	Allow continuing with the next rows if this field has an invalid value.
COLUMN_x_IGNORE_MISSING_COLUMN	Boolean	If the column position will be find in the header line this flag allows this column to be missing.
COLUMN_x_LOCALE	String	To interpret numbers correctly the component needs to know the country or language. Set here the ISO 2-letter or 5-letter country code. Examples: en or en_UK or de_DE
COLUMN_x_NAME	String	The name of the column in the header. Also if no header search in intended the name is mandatory. As name is also possible a regex expression if the component have to search the column in the header by the help of regularly expressions.
COLUMN_x_ALTERNATIVE_FIELD	String	The name of the column which value have to take if the own value id not filled.
COLUMN_x_NULL_ENABLED	Boolean	Null value allowed of not
COLUMN_x_POSITIONTYPE	Integer	Absolute position = 0 Delimited = 2 Delimited with max. length = 3
COLUMN_x_TRIM	Boolean	Trim the value if true. Regardless of this setting empty values (number white spaces == 0) will always returned as null!
CHARSET	String	The charset of the file. Typical values are: UTF-8, UTF-16, Cp-1252, ISO-8859-15, ASCII
ALLOW_ENCLUSURE_IN_TEXT	Boolean	If enclosures are used this flag allows enclosures also in the text it self.
DELIMITER	char	The delimiter character
ENCLOSURE	char	The character used to enclose the field content. Especially useful of the delimiter can also be part of the content or if the content contains line breaks.
IGNORE_ENCLOSED_LINE_BREAK	Boolean	If the content can contains line breaks and is enclosed, set this true
IGNORE_NOT_NULL_CONSTRAINTS	Boolean	true = switch off the check of nullable or not.
SKIP_EMPTY_LINES	Boolean	Empty lines will be skipped
SKIP_ROWS	Integer	Skip number of lines in the file before start parsing it. The header-line will be taken after skipping lines.

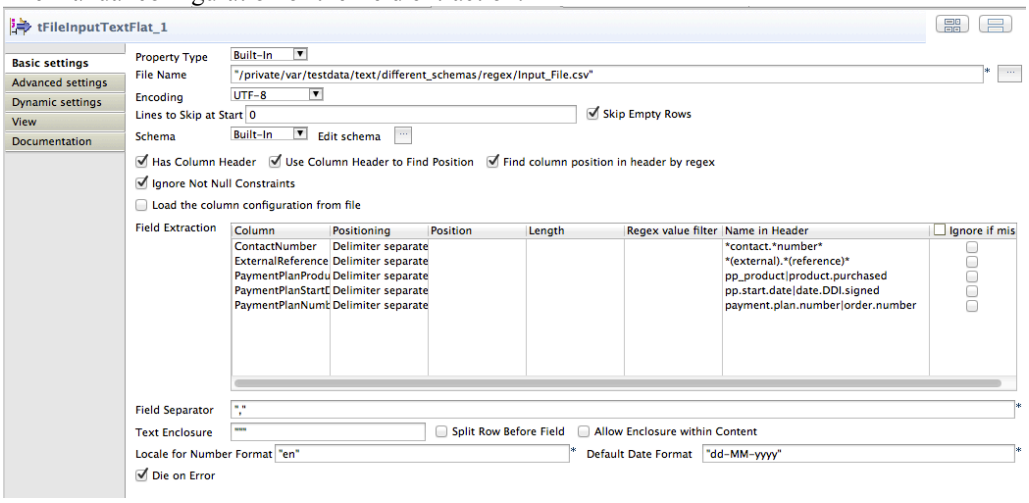
These are the possible classes:

Data class	Corresponding basic type
String	0
Date	1
BigDecimal	2
Long	2
Integer	2
Double (default)	2
Float	2
Short	2
Boolean	8

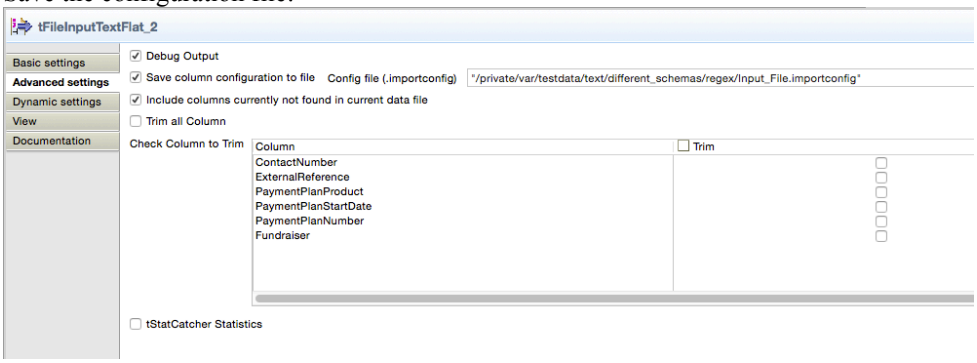
Scenario for loading the configuration from an external file

At first to create the configuration file (as an template for your own changes) you can use the component with a job specific field configuration and if it works you can save it in the advanced settings.
The configuration file can be created in the advanced settings (only if the Load-option is switched off).

The manual configuration of the field extraction:



Save the configuration file:

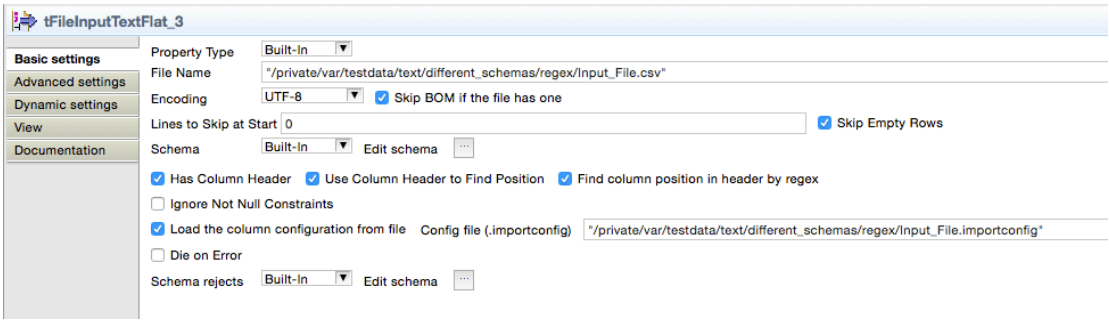


Here the configuration file matching to the manual configuration above.

```
ALLOW_ENCLUSURE_IN_TEXT=false
CHARSET=UTF-8
COLUMN_0_BASICTYPE=0
COLUMN_0_CLASS=String
COLUMN_0_DEFAULT=99
COLUMN_0_DELIMITERCOUNT=0
COLUMN_0_ENABLED=true
COLUMN_0_IGNORE_DATASET_IF_INVALID=false
COLUMN_0_IGNORE_MISSING_COLUMN=false
COLUMN_0_LOCALE=en_US
COLUMN_0_NAME=*contact.*number*
COLUMN_0_NULL_ENABLED=true
COLUMN_0_POSITIONTYPE=2
COLUMN_0_TRIM=false
```

```
COLUMN_1_BASICTYPE=0
COLUMN_1_CLASS=String
COLUMN_1_DELIMITERCOUNT=1
COLUMN_1_ENABLED=true
COLUMN_1_IGNORE_DATASET_IF_INVALID=false
COLUMN_1_IGNORE_MISSING_COLUMN=false
COLUMN_1_LOCALE=en_US
COLUMN_1_NAME=*(external).*(reference)*
COLUMN_1_NULL_ENABLED=true
COLUMN_1_POSITIONTYPE=2
COLUMN_1_TRIM=false
COLUMN_2_BASICTYPE=1
COLUMN_2_CLASS=Date
COLUMN_2_DELIMITERCOUNT=8
COLUMN_2_ENABLED=true
COLUMN_2_FORMAT=dd.MM.yyyy
COLUMN_2_IGNORE_DATASET_IF_INVALID=false
COLUMN_2_IGNORE_MISSING_COLUMN=false
COLUMN_2_LOCALE=en_US
COLUMN_2_NAME=pp.start.date|date.DDI.signed
COLUMN_2_NULL_ENABLED=true
COLUMN_2_POSITIONTYPE=2
COLUMN_2_TRIM=false
COLUMN_3_BASICTYPE=2
COLUMN_3_CLASS=Long
COLUMN_3_DELIMITERCOUNT=71
COLUMN_3_ENABLED=true
COLUMN_3_FORMAT=en
COLUMN_3_IGNORE_DATASET_IF_INVALID=false
COLUMN_3_IGNORE_MISSING_COLUMN=false
COLUMN_3_LOCALE=en_US
COLUMN_3_NAME=payment.plan.number|order.number
COLUMN_3_NULL_ENABLED=true
COLUMN_3_POSITIONTYPE=2
COLUMN_3_TRIM=false
COLUMN_4_BASICTYPE=0
COLUMN_4_CLASS=String
COLUMN_4_DELIMITERCOUNT=45
COLUMN_4_ENABLED=true
COLUMN_4_IGNORE_DATASET_IF_INVALID=false
COLUMN_4_IGNORE_MISSING_COLUMN=false
COLUMN_4_LOCALE=en_US
COLUMN_4_NAME=pp_product|product.purchased
COLUMN_4_NULL_ENABLED=true
COLUMN_4_POSITIONTYPE=2
COLUMN_4_TRIM=false
COLUMN_5_BASICTYPE=0
COLUMN_5_CLASS=String
COLUMN_5_DELIMITERCOUNT=4
COLUMN_5_ENABLED=true
COLUMN_5_IGNORE_DATASET_IF_INVALID=false
COLUMN_5_IGNORE_MISSING_COLUMN=false
COLUMN_5_LOCALE=en_US
COLUMN_5_NAME=Fundraiser
COLUMN_5_NULL_ENABLED=true
COLUMN_5_POSITIONTYPE=2
COLUMN_5_TRIM=false
DELIMITER=,
ENCLOSURE="
IGNORE_BOM=false
IGNORE_ENCLOSED_LINE_BREAK=true
IGNORE_NOT_NULL_CONSTRAINTS=false
SKIP_EMPTY_LINES=true
SKIP_ROWS=0
```

Now switch on the Load option (as in the screenshot above) and it should work in the same way.



Please take care the column data type in the configuration file is the same as in the schema you have set in the job! The problem here is: the data type from the configuration will be used to build the value and the schema type could otherwise probably not matched to the read data.