tFileInputKeyValue

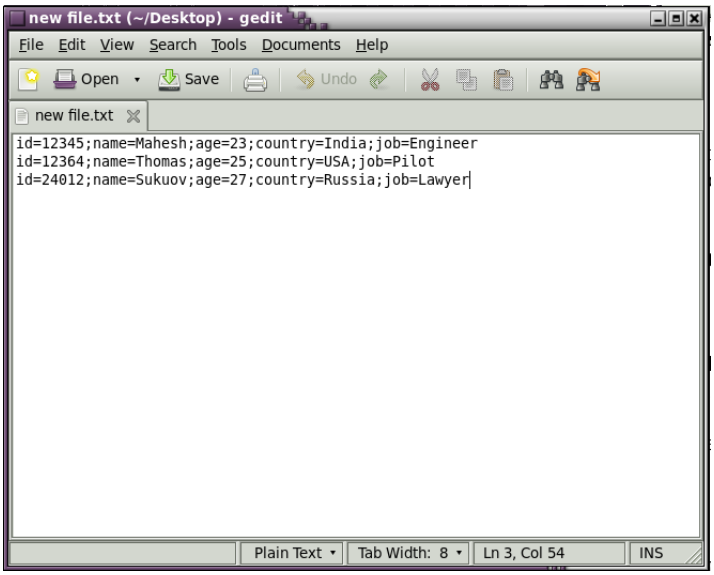
tFileInputKeyValue properties

Component family	File/Input	
Function	**tFileInputKeyValue** reads a given file containing key-value pairs, row by row with separated fields.	
Purpose	Opens a file containing “key-value” pairs in each row ,and extracts the “values” from the “keys”.The “keys” are taken as fields and “values” as its contents. (This component allows empty key-value pairs in the input). It then sends the extracted values as defined in the Schema to the next Job component, via a Row link.	
Basic settings	*File Name*	**File name**: Name and path of the file to be processed. In order to avoid the inconvenience of hand writing, you could select the variable of interest from the auto-completion list (Ctrl +Space) to fill the current field on condition that this variable has been properly defined.
	Field Separator	It is used to seperate key-value pairs in each row. It has to be a Regex Expression.
	Row Separator	It is used to distinguish rows. It has to be a Regex Expression.
	Key-Value Separator	It separates a Value from a key. It has to be a Regex Expression.
	Key Name	**Column**: This field is automatically populated with the columns defined in the schema that you propagated.
		Key:This field should contain the “key” names as of in the input file.
		Trim Value: Select this check box to remove leading and trailing whitespaces from defined columns.
	Header	Number of rows to be skipped in the beginning of file
	Footer	Number of rows to be skipped at the end of the file.
	Limit	Maximum number of rows to be processed. If Limit = 0, no row is read or processed.
	Schema and Edit Schema	A schema is a row description, i.e., it defines the number of fields that will be processed and passed on to the next component. The schema is either built-in or remote in the Repository.
		Built-in: The schema will be created and stored locally for this component only.
		Repository: The schema already exists and is stored in the Repository, hence can be reused in various projects and Job flowcharts.

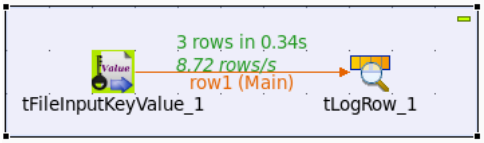
	Die on error	Select this check box to stop the execution of the Job when an error occurs. Clear the check box to skip the row on error and complete the process for error-free rows.
Advanced settings	*tStatCatcher Statistics*	Select this check box to gather the processing metadata at the Job level as well as at each component level.
Usage	Use this component to read a file containing “key-value” pairs in each row . It extracts the “values” from the “keys”. The “keys” are taken as fields and “values” as its contents. (This component even allows empty key-value pairs in the input). It then sends the extracted values as defined in the Schema to the next Job component, via a Row link. For further information, please see the section called “Scenario 2: Extracting values from a file containing key value pairs”.	
Author	Mahesh M.Pillai	

Scenario 1: Reading data from a file containing multiple Key-Value Pairs in a Row

The following scenario creates a two-component Job, which aims at reading each row (containing multiple key-value pair) of a file ,extracts the “value” from the “keys ” and displays the output in the Run log console.
The contents of the Input Text File is shown below:

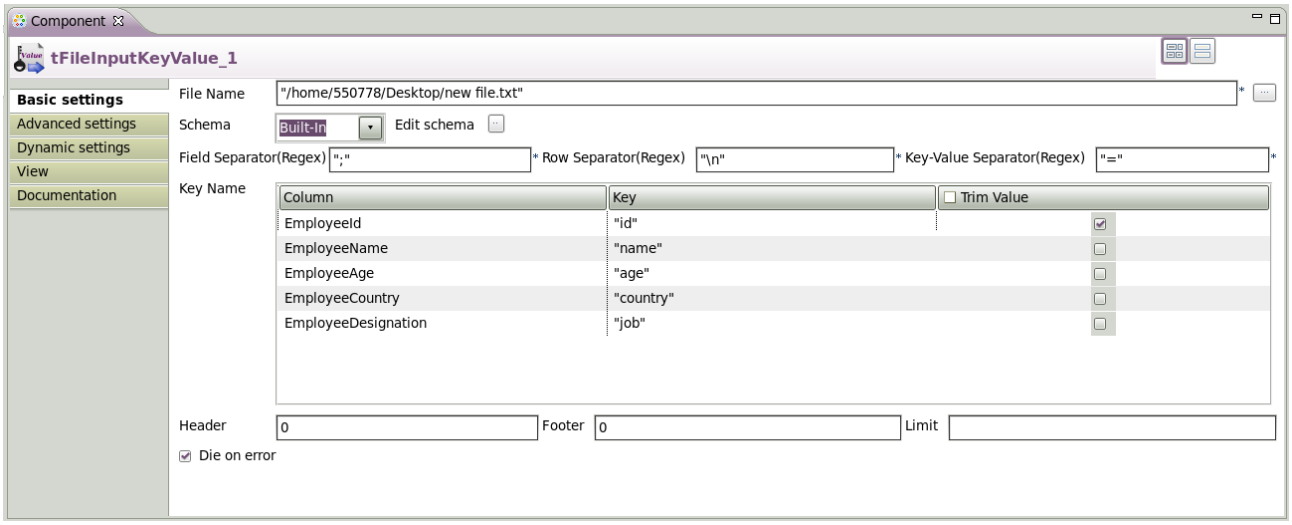


• Here the keys are “id”, “name”, “age”, “country”, “job”.

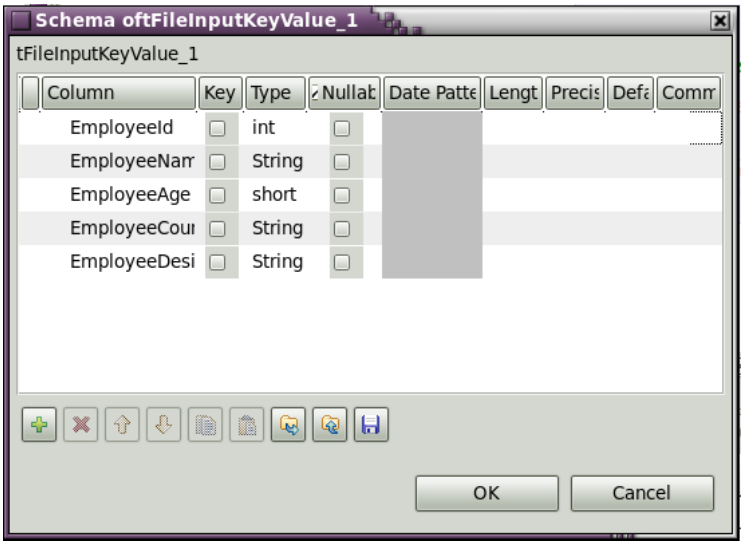


• Drop a **tFileInputKeyValue** component from the Palette to the design workspace.
• Drop a **tLogRow** component the same way.
• Right-click on the **tFileInputKeyValue** component and select **Row** > **Main**. Then drag it onto the **tLogRow** component and release when the plug symbol shows up.

• Select the **tFileInputKeyValue** component again, and define its **Basic** settings:



• Fill in a path to the file in the **File Name** field. This field is mandatory.
If the path of the file contains some accented characters, you will get an error message when executing your Job. For more information regarding the procedures to follow when the support of accented characters is missing, see Talend Open Studio for Big Data Installation Guide.
• Set the **Schema** as either a local (Built-in) or a remotely managed (Repository) to define the data to pass on to the **tLogRow** component.
• You can load and/or edit the schema via the **Edit Schema** function.
In the present scenario we have in total a maximum of 5 key-value pairs in a row.
You may define them as shown below.

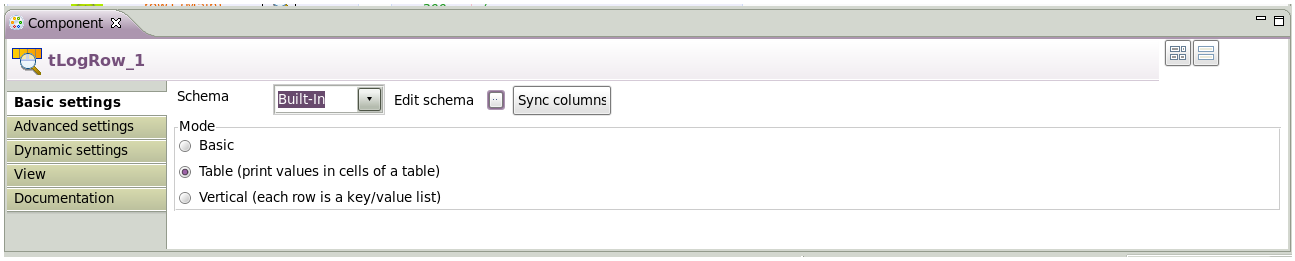


• The column name can be any valid name.
• Define the **Field separator** used to delimit key-value pairs in a row (Here it is ";").Then define the **Row Separator** allowing to identify the end of a row (Here it is "\n"). Also define the **Key-Value Separator** (Here it is "=").
These fields should be in Regex Patterns.
• Now the **Key Name** Field should be filled.
Column field is automatically populated with the columns defined in the schema that you propagated.
The **Key** field corresponds to the keys in the file that we have opened.

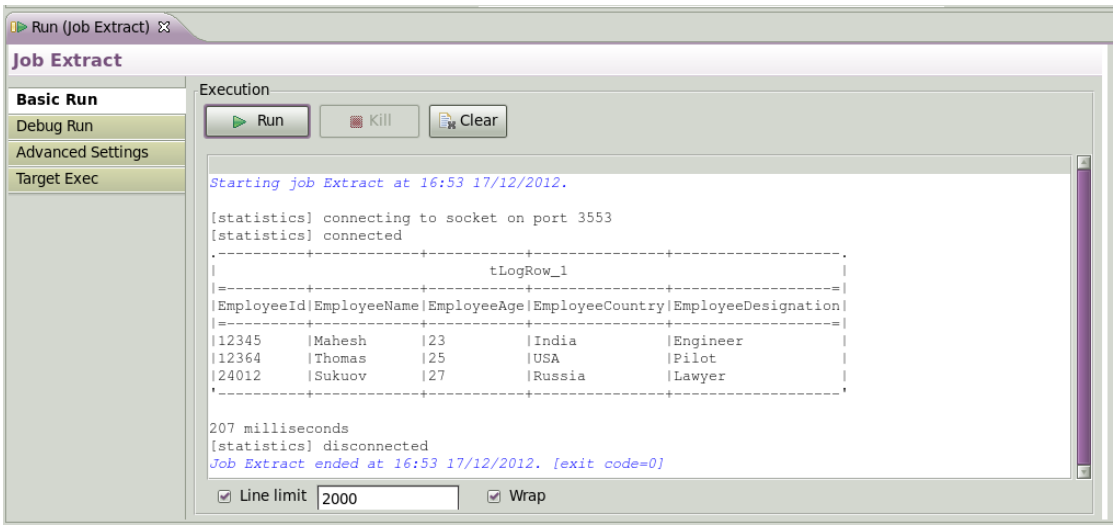
In the current scenario the keys are “id”, “name”, “age”, “country”, “job”.
The **Trim Value** field can be used to remove leading and trailing whitespaces from the values corresponding to the fields.

• In this scenario, the **Header** , **Footer** , **Limit** numbers are not set.
•Die on Error can be set as per need.(Select this check box to stop the execution of the Job when an error occurs. Clear the check box to skip the row on error and complete the process for error-free rows.)

• Select the tLogRow and define the Field separator to use for the output display.



• Go to **Run** tab, and click on Run to execute the Job.
The file is read row by row and the extracted fields are displayed on the Run log as defined in both components Basic settings.



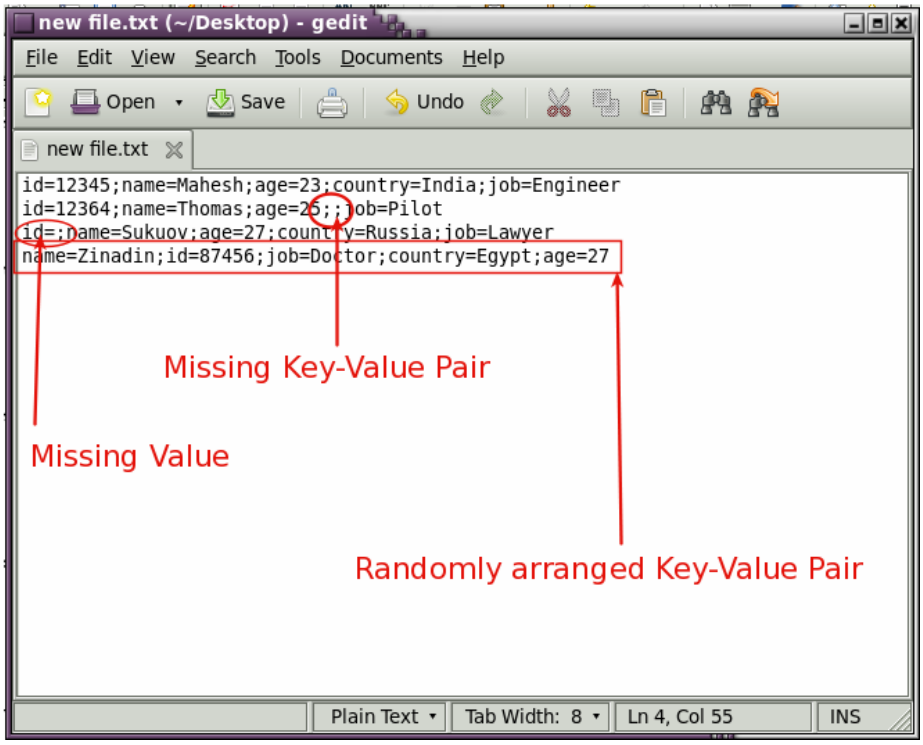
Scenario 2: Reading data from a file that contains

- **Key-Value Pairs randomly distributed in a row**
- **Key-Value Pairs missing in a row**

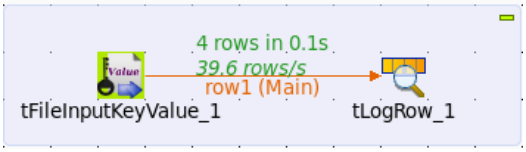
*The following scenario describes the **Flexibility** this component offers:*

- *It allows missing key value pairs.*
- *It allows missing values.*
- *It allows random distribution of key-value pairs within the row.*

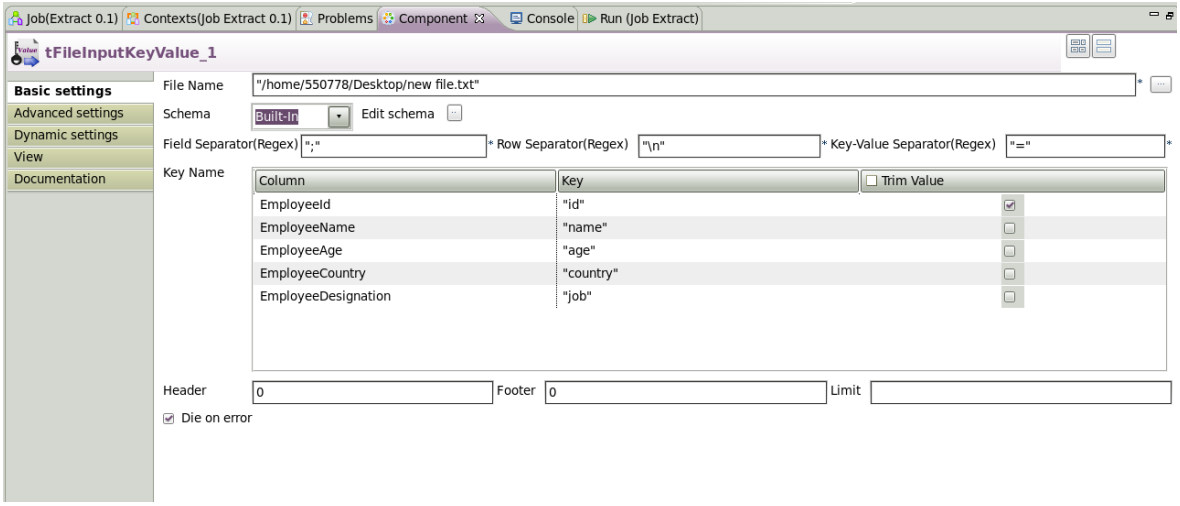
The following scenario depicts a two-component Job, which aims at reading each row (containing multiple key-value pair) of a file ,extracts the “value” from the “keys ” and displays the output in the Run log console.
The above mentioned(*in italics)* conditions are created in the input file.
The Input File (with Annotations) is shown below:



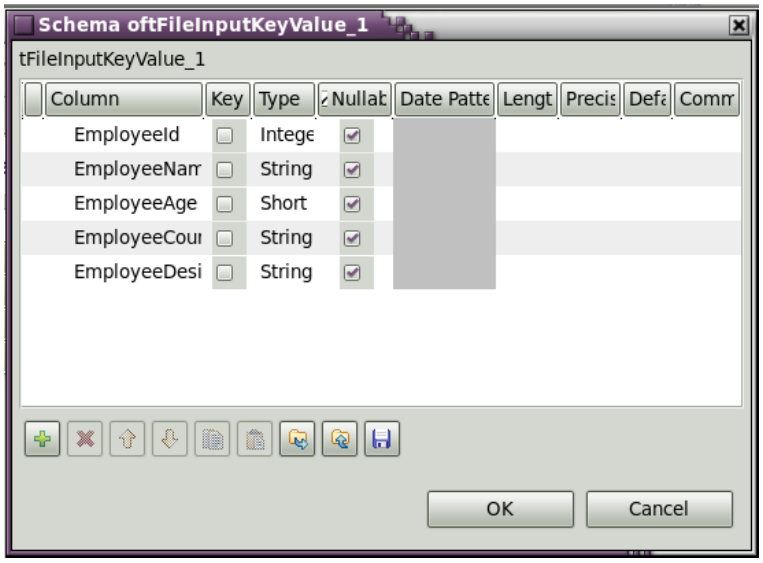
• Drop a **tFileInputKeyValue** component from the Palette to the design workspace.
• Drop a **tLogRow** component the same way.
• Right-click on the **tFileInputKeyValue** component and select **Row** > **Main**. Then drag it onto the **tLogRow** component and release when the plug symbol shows up.



• Select the **tFileInputKeyValue** component again, and define its **Basic** settings:



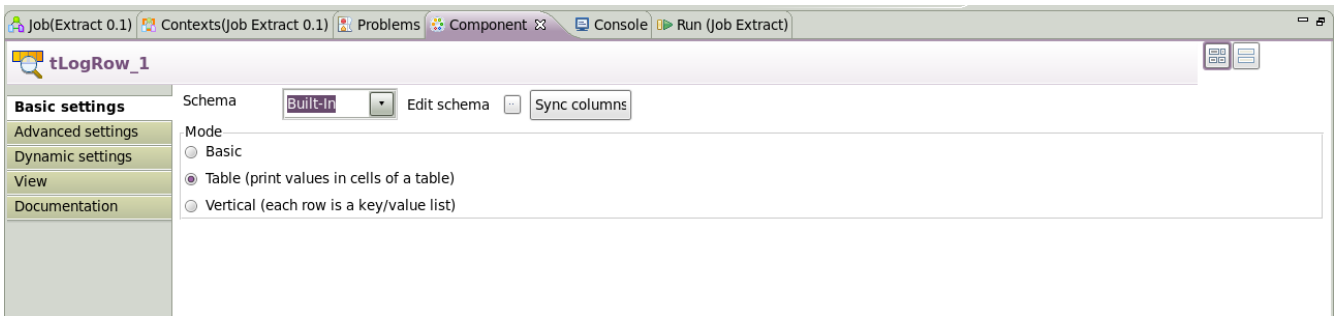
• Fill in a path to the file in the **File Name** field. This field is mandatory.
If the path of the file contains some accented characters, you will get an error message when executing your Job. For more information regarding the procedures to follow when the support of accented characters is missing, see Talend Open Studio for Big Data Installation Guide.
• Set the **Schema** as either a local (Built-in) or a remotely managed (Repository) to define the data to pass on to the **tLogRow** component.
• You can load and/or edit the schema via the **Edit Schema** function.
In the present scenario we have in total a maximum of 5 key-value pairs in a row.
You may define them as shown below.



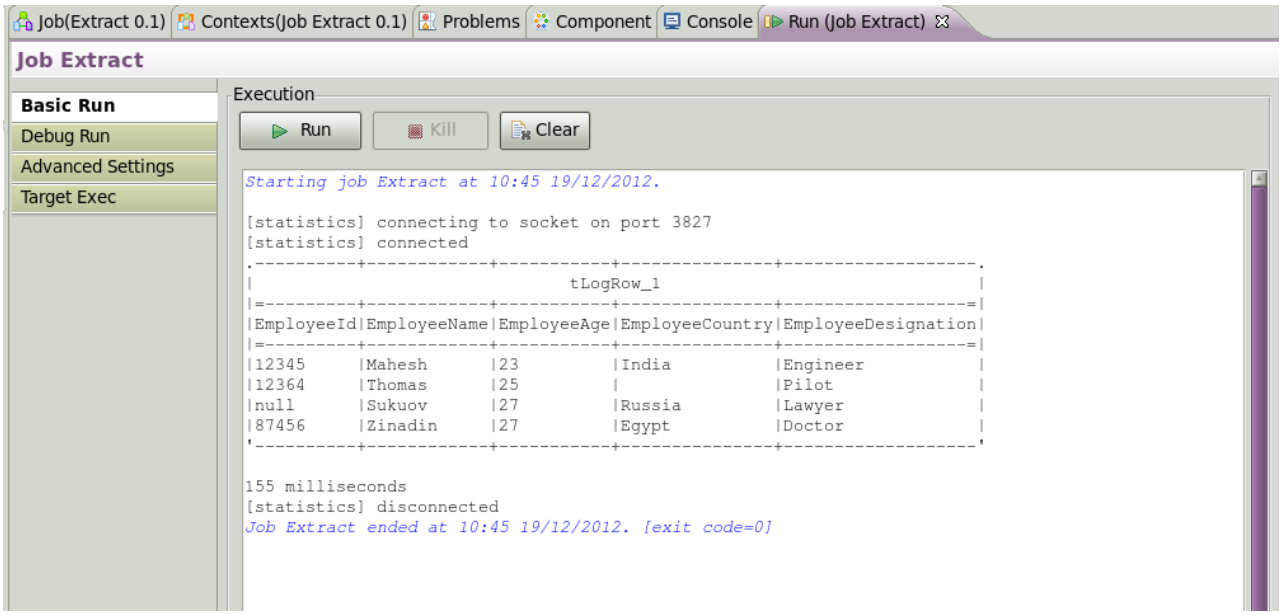
• The column name can be any valid name.
• Define the **Field separator** used to delimit key-value pairs in a row (Here it is ";").Then define the **Row Separator** allowing to identify the end of a row (Here it is "\n"). Also define the **Key-Value Separator** (Here it is "=").
These fields should be in Regex Patterns.
• Now the **Key Name** Field should be filled.
Column field is automatically populated with the columns defined in the schema that you propagated.
The **Key** field corresponds to the keys in the file that we have opened.
In the current scenario the keys are "id", "name", "age", "country", "job".
The **Trim Value** field can be used to remove leading and trailing whitespaces from the values corresponding to the fields.

• In this scenario, the **Header** , **Footer** , **Limit** numbers are not set.
•Die on Error can be set as per need.(Select this check box to stop the execution of the Job when an error occurs. Clear the check box to skip the row on error and complete the process for error-free rows.)

• Select the tLogRow and define the Field separator to use for the output display.



• Go to **Run** tab, and click on Run to execute the Job.
The file is read row by row and the extracted fields are displayed on the Run log as defined in both components Basic settings.



Inspite of the input being in an unordered format, the output is in the correct format.