

# AtlasMap User Guide

The AtlasMap Team

Version 1.40.0-SNAPSHOT, 2019-03-06

# AtlasMap User Guide

|  |   |
|--|---|
| 1. Quickstart .....  | 2 |
| 1.1. Running AtlasMap Data Mapper UI standalone .....                              | 2 |
| 1.2. Running AtlasMap data mapping with Apache Camel .....                         | 2 |
| 2. Design Mappings in UI .....   | 3 |
| 2.1. Find the data field that you want to map .....                                | 3 |
| 2.2. Map one source field to one target field .....                                | 3 |
| 2.3. Example of missing or unwanted data when combining or separating fields ..... | 4 |
| 2.4. Combine multiple source fields into one target field .....                    | 5 |
| 2.5. Separate one source field into multiple target fields .....                   | 7 |
| 2.6. Transform source or target data .....   | 8 |
| 2.7. View the mappings .....   | 8 |
| 2.8. Descriptions of available transformations .....                               | 9 |

AtlasMap is a data mapping solution with an interactive web based user interface. It simplifies configuration of an integration that handles different types of data, such as Java, XML, or JSON data. You design your data mapping in the AtlasMap Data Mapper UI canvas, and then run that data mapping by means of a runtime engine. We have a camel-atlasmap component which consumes AtlasMap mapping definition and process data mapping as a part of Camel route.

# Chapter 1. Quickstart

## 1.1. Running AtlasMap Data Mapper UI standalone

Here is the shortest path to run standalone AtlasMap.

1. Download AtlasMap standalone jar

```
wget http://central.maven.org/maven2/io/atlasmap/atlasmap-standalone/${VERSION}/atlasmap-standalone-${VERSION}.jar
```

2. Run AtlasMap standalone

```
$ java -jar atlasmap-standalone-${VERSION}.jar
```

Now AtlasMap Data Mapper UI is available at <http://127.0.0.1:8585/>

## 1.2. Running AtlasMap data mapping with Apache Camel



TODO

camel-atlasmap endpoint use IN body as a default source document of mappings. If IN body is a `java.util.Map`, key is used as a Document ID and corresponding value is used as a Document payload.

```
...
from("direct:start")
  .to("atlas:atlas-mapping.adm")
  .log("${body}")
...
```

# Chapter 2. Design Mappings in UI

## 2.1. Find the data field that you want to map


In a relatively simple integration, mapping data fields is easy and intuitive. In more complex integrations or in integrations that handle large sets of data fields, mapping from source to target is easier when you have some background about how to use the data mapper.

The data mapper displays two columns of data fields:

- **Sources** is a list of the data fields that you can map to target fields.
- **Target** is a list of the data fields that you can map source fields to.

To quickly find the data field that you want to map, you can do any of the following:

- Search for it.

The **Sources** panel and the **Target** panel each have a search field at the top. If the search field is not visible, click  at the top right of the **Sources** or **Target** panel.

- Enter the names of the fields that you want to map.

To do this, in the upper right of the **Configure Mapper** page, click the plus sign to display the **Mapping Details** panel. In the **Sources** section, enter the name of the source field. In the **Action** section, accept the default **Map**, which maps one field to one other field. Or, select **Combine** or **Separate**. In the **Target** section, enter the name of the field that you want to map to.

- Expand and collapse folders to limit the visible fields.


## 2.2. Map one source field to one target field

The default mapping behavior maps one source field to one target field. For example, map the **Name** field to the **CustomerName** field.

### *Procedure*

1. In the **Sources** panel, click the data field that you want to map from.

You might need to expand a folder to see the data fields that it provides.

When there are many source fields, you can search for the field of interest by clicking the  and entering the name of the data field in the search field.


2. In the **Target** panel, click the data field that you want to map to.


The data mapper displays a line that connects the two fields that you just selected.

3. Optionally, preview the data mapping result. This is useful when you add a transformation to the mapping or when the mapping requires a type conversion.

- a. In the upper right of the data mapper, click  and select **Show Mapping Preview** to

display a text input field on the source field and a read-only result field on the target field.

- b. In the source field's data input field, enter text. Click somewhere outside this text box to display the mapping result in the read-only field on the target field.
  - c. Optionally, to see the result of a transformation, add a transformation in the **Mapping Details** panel.
  - d. Hide the preview fields by clicking [Editor settings] again and selecting **Show Mapping Preview**.
4. Optionally, to confirm that the mapping is defined, in the upper right, click  to display the defined mappings.

You can also preview data mapping results in this view. If preview fields are not visible, click  and select **Show Mapping Preview**. Enter data as described in the previous step. In the table of defined mappings, preview fields appear for only the selected mapping. To see preview fields for another mapping, select it.

Click  again to display the data field panels.

5. In the upper right, click **Done** to save the mapping.

#### Alternative procedure

Here is another way to map a single source field to a single target field:

1. In the **Configure Mapper** page, in the upper right, click the plus sign to display the **Mapping Details** panel.
2. In the **Sources** section, enter the name of the source field.
3. In the **Action** section, accept the default **Map** action.
4. In the **Target** section, enter the name of the field that you want to map to and click **Enter**.

## 2.3. Example of missing or unwanted data when combining or separating fields

In a data mapping, you might need to identify missing or unwanted data when a source or target field contains compound data. For example, consider a **long\_address** field that has this format:

*number street apartment city state zip zip+4 country*

Suppose that you want to separate the **long\_address** field into discrete fields for **number**, **street**, **city**, **state**, and **zip**. To do this, you select **long\_address** as the source field and then select the target fields. You then add padding fields at the locations for the parts of the source field that you do not want. In this example, the unwanted parts are *apartment*, *zip+4*, and *country*.

To identify the unwanted parts, you need to know the order of the parts. The order indicates an index for each part of the content in the compound field. For example, the **long\_address** field has 8 ordered parts. Starting at 1, the index of each part is:

|   |               |
|---|---------------|
| 1 | <i>number</i> |
|---|---------------|

|   |                  |
|---|------------------|
| 2 | <i>street</i>    |
| 3 | <i>apartment</i> |
| 4 | <i>city</i>      |
| 5 | <i>state</i>     |
| 6 | <i>zip</i>       |
| 7 | <i>zip+4</i>     |
| 8 | <i>country</i>   |

In the data mapper, to identify *apartment*, *zip+4*, and *country* as missing, you add padding fields at indexes 3, 7, and 8. See [Separate one source field into multiple target fields](#).

Now suppose that you want to combine source fields for *number*, *street*, *city*, *state*, and *zip* into a *long\_address* target field. Further suppose that there are no source fields to provide content for *apartment*, *zip+4*, and *country*. In the data mapper, you need to identify these fields as missing. Again, you add padding fields at indexes 3, 7, and 8. See [Combine multiple source fields into one target field](#).

## 2.4. Combine multiple source fields into one target field

In a data mapping, you can combine multiple source fields into one compound target field. For example, you can map the *FirstName* and *LastName* fields to the *CustomerName* field.

### Prerequisite

For the target field, you must know what type of content is in each part of this compound field, the order and index of each part of the content, and the separator between parts, such as a space or comma. See [Example of missing or unwanted data when combining or separating fields](#).

### Procedure

1. In the **Target** panel, click the field into which you want to map more than one source field.
2. In the **Sources** panel, if there is a field that contains the fields that you want to map to the target field, then click that container field to map all contained fields to the target field.

To individually select each source field, click the first field that you want to combine into the target field. For each of the other fields that you want to combine into the target field, hover over that field, and press **CTRL-Mouse1** (**CMD-Mouse1** on MacOS).

The data mapper automatically changes the field action from **Map** to **Combine**.

When you are done you should see a line from each of the source fields to the target field.

3. In the **Mapping Details** panel, in the **Separator** field, accept or select the character that the data mapper inserts in the target field between the content from different source fields. The default is a space.
4. In the **Mapping Details** panel, under **Sources**, ensure that the source fields are in the same


order as the corresponding content in the compound target field.

If necessary, drag and drop source fields to achieve the same order. The data mapper automatically updates the index numbers to reflect the new order.

5. If you mapped a source field to each part of the compound target field, then skip to the next step.

If the target field expects data that is not available to be mapped, then in the **Mapping Details** panel, edit the index of each source field so that it is the same as the index of the corresponding data in the compound target field. The data mapper automatically adds padding fields as needed to indicate missing data.


If you accidentally create too many padding fields, click the trash icon on each extra padding field to delete it.

6. Optionally, preview the data mapping result:
  - a. In the upper right of the data mapper, click  and select **Show Mapping Preview** to display a text input field on each source field for the currently selected mapping and a read-only result field on the target field of the currently selected mapping.
  - b. In the source data input fields, enter text. Click outside the text box to display the mapping result in the read-only field on the target field.




If you reorder the source fields or add a transformation to the mapping then the result field on the target field reflects this. If the data mapper detects any errors, it displays informative messages at the top of the **Mapping Details** panel.


- c. Hide the preview fields by clicking  again and selecting **Show Mapping Preview**.

If you redisplay the preview fields, any data that you entered in them is still there and it remains there until you exit the data mapper.

7. To confirm that the mapping is correctly defined, in the upper right, click  to display the defined mappings. A mapping that combines the values of more than one source field into one target field looks like this:

## Mappings

|  Sources |  Targets |  Type |
|---|---|--|
| /FirstName ⚡  | /first_and_last_name  | Combine  |
| /LastName ⚡   |   | (Space [ ]) .  |

You can also preview mapping results in this view. Click , select **Show Mapping Preview**, and enter text as described in the previous step. Preview fields appear for only the selected mapping. Click another mapping in the table to view preview fields for it.



## 2.5. Separate one source field into multiple target fields

In a data mapping, you can separate a compound source field into multiple target fields. For example, map the **Name** field to the **FirstName** and **LastName** fields.

### *Prerequisite*

For the source field, you must know what type of content is in each part of this compound field, the order and index of each part of the content, and the separator between parts, such as a space or comma. See [Example of missing or unwanted data when combining or separating fields](#).

### *Procedure*

1. In the **Sources** panel, click the field whose content you want to separate.
2. In the **Target** panel, click the first field that you want to separate the source field data into.
3. In the **Target** panel, for each additional target field that you want to contain some of the data from the source field, hover over the field and press **CTRL-Mouse1** (**CMD-Mouse1** on MacOS) to select it.

The data mapper automatically changes the field action to **Separate**.


When you are done selecting target fields, you should see lines from the source field to each of the target fields.

4. In the **Mapping Details** panel, in the **Separator** field, accept or select the character in the source field that indicates where to separate the source field values. The default is a space.
5. In the **Mapping Details** panel, under **Targets**, ensure that the target fields are in the same order as the corresponding content in the compound source field.

If necessary, drag and drop target fields to achieve the same order. The data mapper automatically updates the index numbers to reflect the new order.

6. If you mapped each part of the compound source field to a target field, then skip to the next step.

If the source field contains data that you do not need, then in the **Mapping Details** panel, edit the index of each target field so that it is the same as the index of the corresponding data in the compound source field. The data mapper automatically adds padding fields as needed to indicate unwanted data.


7. Optionally, preview the data mapping result:
  - a. In the upper right of the data mapper, click  and select **Show Mapping Preview** to display a text input field on the source field and read-only result fields on each target field.
  - b. In the source field's data input field, enter text. Be sure to enter the separator character between the parts of the field. Click outside the text box to display the mapping result in the read-only fields on the target fields.


If you reorder the target fields or add a transformation to a target field then the result fields

on the target fields reflect this. If the data mapper detects any errors, it displays informative messages at the top of the **Mapping Details** panel.

- c. Hide the preview fields by clicking  again and selecting **Show Mapping Preview**.

If you redisplay the preview fields, any data that you entered in them is still there and it remains there until you exit the data mapper.

8. To confirm that the mapping is correctly defined, click  to display defined mappings. A mapping that separates the value of a source field into multiple target fields looks like this: [Separate Fields Mapping].

You can also preview mapping results in this view. Click , select **Show Mapping Preview**, and enter text as described in the previous step. Preview fields appear for only the selected mapping. Click another mapping in the table to view preview fields for it.

## 2.6. Transform source or target data

In the data mapper, after you define a mapping, you can transform any field in the mapping. Transforming a data field defines how you want to store the data. For example, you could specify the **Capitalize** transformation to ensure that the first letter of a data value is uppercase.

### *Procedure*

1. Map the fields. This can be a one-to-one mapping, a combination mapping, or a separation mapping.
2. In the **Mapping Details** panel, under **Sources** or under **Targets**, in the box for the field that you want to transform, click the arrow that points to the trash can. This displays a field where you can select the transformation that you want the data mapper to perform.
3. Click in this field to display the list of transformations.
4. Click the transformation that you want to perform.
5. If the transformation requires any input parameters, specify them in the appropriate input fields.
6. To add another transformation, click the arrow that points to the trash can again.

### *Additional resource*

[Descriptions of available transformations](#)



## 2.7. View the mappings

While you are using the data mapper UI, you can view the mappings that are already defined. This lets you check whether the correct mappings are in place.

### *Prerequisites*

The data mapper canvas is visible.

### *Procedure*

1. In the upper right, click  to display a list of the defined mappings.
2. To dismiss the list of mappings and redisplay the source and target fields, click  again.

## 2.8. Descriptions of available transformations



TODO Generate this list automatically from annotation - <https://github.com/atlasmap/atlasmap/issues/173>

The following table describes the available transformations. The date and number types refer generically to any of the various forms of these concepts. That is, number includes, for example, *integer*, *long*, *double*. Date includes, for example, *date*, *Time*, *ZonedDateTime*.

| Transformation       | Input Type | Output Type | Parameter (* = required) | Description  |
|----------------------|------------|-------------|--------------------------|--|
| <i>AbsoluteValue</i> | number     | number      | None                     | Return the absolute value of a number.   |
| <i>AddDays</i>       | date       | date        | <i>days</i>              | Add days to a date. The default is 0 days.   |
| <i>AddSeconds</i>    | date       | date        | <i>seconds</i>           | Add seconds to a date. The default is 0 seconds.   |
| <i>Append</i>        | string     | string      | string                   | Append a string to the end of a string. The default is to append nothing.  |
| <i>Camelize</i>      | string     | string      | None                     | Convert a phrase to a camelized string by removing whitespace, making the first word lowercase, and capitalizing the first letter of each subsequent word. |
| <i>Capitalize</i>    | string     | string      | None                     | Capitalize the first character in a string.  |
| <i>Ceiling</i>       | number     | number      | None                     | Return the whole number ceiling of a number.   |
| <i>Contains</i>      | any        | Boolean     | <i>value</i>             | Return true if a field contains the specified value.   |

| Transformation      | Input Type | Output Type | Parameter (* = required) | Description   |
|---------------------|------------|-------------|--------------------------|---|
| ConvertAreaUnit     | number     | number      | fromUnit*<br>toUnit *    | Convert a number that represents an area to another unit. For the fromUnit and toUnit parameters, select the appropriate unit from the <b>From Unit</b> and <b>To Unit</b> menus. The choices are: Square Foot, Square Meter, or Square Mile. |
| ConvertDistanceUnit | number     | number      | fromUnit*<br>toUnit *    | Convert a number that represents a distance to another unit. For the fromUnit and toUnit parameters, select the appropriate unit from the <b>From Unit</b> and <b>To Unit</b> menus. The choices are: Foot, Inch, Meter, Mile, or Yard.       |
| ConvertMassUnit     | number     | number      | fromUnit*<br>toUnit *    | Convert a number that represents mass to another unit. For the fromUnit and toUnit parameters, select the appropriate unit from the <b>From Unit</b> and <b>To Unit</b> menus. The choices are: Kilogram or Pound.                            |

| Transformation    | Input Type | Output Type | Parameter (* = required) | Description   |
|-------------------|------------|-------------|--------------------------|---|
| ConvertVolumeUnit | number     | number      | fromUnit *<br>toUnit *   | Convert a number that represents volume to another unit. For the fromUnit and toUnit parameters, select the appropriate unit from the <b>From Unit</b> and <b>To Unit</b> menus. The choices are: Cubic Foot, Cubic Meter, Gallon US Fluid, or Liter. |
| CurrentDate       | None       | date        | Note                     | Return the current date.  |
| CurrentDateTime   | None       | date        | None                     | Return the current date and time.   |
| CurrentTime       | None       | date        | None                     | Return the current time.  |
| DayOfWeek         | date       | number      | None                     | Return the day of the week (1 through 7) that corresponds to the date.  |
| DayOfYear         | date       | number      | None                     | Return the day of the year (1 through 366) that corresponds to the date.  |
| EndsWith          | string     | Boolean     | string                   | Return true if a string ends with the specified string, including case.   |
| Equals            | any        | Boolean     | value                    | Return true if a field is equal to the specified value, including case.   |
| FileExtension     | string     | string      | None                     | From a string that represents a file name, return the file extension without the dot.   |

| Transformation | Input Type | Output Type | Parameter (* = required) | Description  |
|----------------|------------|-------------|--------------------------|--|
| Floor          | number     | number      | None                     | Return the whole number floor of a number.   |
| Format         | any        | string      | template *               | In <b>template</b> , replace each placeholder (such as %s) with the value of the input field and return a string that contains the result. This is similar to mechanisms that are available in programming languages such as Java and C. |
| GenerateUUID   | None       | string      | None                     | Create a string that represents a random UUID.   |
| IndexOf        | string     | number      | string                   | In a string, starting at 0, return the first index of the specified <b>string</b> . Return -1 if it is not found.  |
| IsNull         | any        | Boolean     | None                     | Return true if a field is null.  |
| LastIndexOf    | string     | number      | string                   | In a string, starting at 0, return the last index of the specified <b>string</b> . Return -1 if it is not found.   |
| Length         | any        | number      | None                     | Return the length of the field, or -1 if the field is null.  |
| Lowercase      | string     | string      | None                     | Convert a string to lowercase.   |

| Transformation | Input Type | Output Type | Parameter (* = required)     | Description   |
|----------------|------------|-------------|------------------------------|---|
| Normalize      | string     | string      | None                         | Replace consecutive whitespace characters with a single space and trim leading and trailing whitespace from a string.                           |
| PadStringLeft  | string     | string      | padCharacter *<br>padCount * | Insert the character supplied in padCharacter at the beginning of a string. Do this the number of times specified in padCount.                  |
| PadStringRight | string     | string      | padCharacter *<br>padCount * | Insert the character supplied in padCharacter at the end of a string. Do this the number of times specified in padCount.                        |
| Prepend        | string     | string      | string                       | Prefix string to the beginning of a string. the default is to prepend nothing.  |
| ReplaceAll     | string     | string      | match *<br>newString         | In a string, replace all occurrences of the supplied matching string with the supplied newString. The default newString is an empty string.     |
| ReplaceFirst   | string     | string      | match *<br>newString *       | In a string, replace the first occurrence of the specified match string with the specified newString. The default newString is an empty string. |

| Transformation       | Input Type | Output Type | Parameter (* = required) | Description  |
|----------------------|------------|-------------|--------------------------|--|
| Round                | number     | number      | None                     | Return the rounded whole number of a number.   |
| SeparateByDash       | string     | string      | None                     | Replace each occurrence of whitespace, colon (:), underscore (_), plus (+), and equals (=) with a hyphen (-).  |
| SeparateByUnderscore | string     | string      | None                     | Replace each occurrence of whitespace, colon (:), hyphen (-), plus (+), and equals (=) with an underscore (_).   |
| StartsWith           | string     | Boolean     | string                   | Return true if a string starts with the specified string (including case).   |
| Substring            | string     | string      | startIndex *<br>endIndex | Retrieve a segment of a string from the specified inclusive <b>startIndex</b> to the specified exclusive <b>endIndex</b> . Both indexes start at zero. <b>startIndex</b> is inclusive. <b>endIndex</b> is exclusive. The default value of <b>endIndex</b> is the length of the string. |



| Transformation  | Input Type | Output Type | Parameter (* = required)            | Description   |
|-----------------|------------|-------------|-------------------------------------|---|
| SubstringAfter  | string     | string      | startIndex *<br>endIndex<br>match * | Retrieve the segment of a string after the specified match string from the specified inclusive startIndex to the specified exclusive endIndex. Both indexes start at zero. The default value of endIndex is the length of the string after the supplied match string. |
| SubstringBefore | string     | string      | startIndex *<br>endIndex<br>match * | Retrieve a segment of a string before the supplied match string from the supplied inclusive startIndex to the supplied exclusive endIndex. Both indexes start at zero. The default value of endIndex is the length of the string before the supplied match string.    |
| Trim            | string     | string      | None                                | Trim leading and trailing whitespace from a string.   |
| TrimLeft        | string     | string      | None                                | Trim leading whitespace from a string.  |
| TrimRight       | string     | string      | None                                | Trim trailing whitespace from a string.   |
| Uppercase       | string     | string      | None                                | Convert a string to uppercase.  |