aaz·app

hello@aaz.app • https://aaz.app

StringWizard

Saved on: Tuesday 14 January 2025 • **Version:** Version 3

Table of Contents

1. Introduction

StringWizard offers string manipulation to change case, replace unwanted characters (such as space, diacritics, ligatures, trademarks or reserved characters), and truncate strings.

This is particularly useful when there is a need of string standardization, such as making filename suitable for the web or making it compatible between platforms (for file transfer or archive).

1.1. Versions

The following is a short version overview:

- **Version 1**: initial version of the app.
- **Version 2**: more diacritics are captured (new method added).
- **Version 3**: datasets can now be converted as well.
  UI/UX of the app has been modified to manage datasets (reorganization of the properties).
  It is now possible to create a custom conversion table.
  The documentation has been updated to list the process private data generated by the app.

1.2. Compatibility

Enfocus Switch 2023 Fall Edition and higher for Windows and Mac.

StringWizard will work on any Switch configuration, but the Metadata module is required to used private data as input and/or output.

2. How to use

StringWizard needs an incoming connection and a single outgoing connection.

StringWizard can perform several string conversions, which can be combined so that multiple changes can be processed simultaneously. You can build your own configuration by enabling or disabling any property of the app.

Modifications are applied in the following order:
Convert diacritics and ligatures > Trademarks > Case > Space > Reserved characters > Custom conversion > Truncate

2.1. String input and output

StringWizard uses different types of input and output to solve many use cases of string conversion, e.g. using a private data input to rename the incoming job, process the incoming jobname and save the result as a private data, convert the content of a private data, convert and replace the content of a specific dataset or create a new dataset from a cleaned existing dataset.

2.1.1. Input type

There are 3 types of input that can be sent to StringWizard:

- **Incoming jobname**
- **Private data**
- **Text**
- **Dataset**

By default, this property is set to ‘Incoming jobname’.

2.1.1.1. Incoming jobname

If the input string is the incoming jobname, another property is requested:

- **Jobname proper**
  Only the jobname proper is processed, the extension is kept as is
- **Complete jobname**
  The complete jobname is processed, including the extension

By default, this property is set to ‘Jobname proper’.

When the input string is the incoming jobname, the output can be:

- **Rename incoming job**
- **Private data**

By default, this property is set to ‘Rename incoming job’.

The output settings are detailed in 2.1.2 Output type.

2.1.1.2. Private data

The input string can be a private data. This private data can be provided inline, using a single-line text with variables or a script expression.

2 properties are requested when the input is a private data:

- **Key**
  The private data key to be used (mandatory)
- **Add incoming job extension on output**
  The incoming job extension can be added or omitted on output (when the output is a renamed job) – default: 'Yes'

When the input string is a private data, the output can be:

- **Rename incoming job**
- **Private data**

By default, this property is set to 'Rename incoming job'.

The output settings are detailed in 2.1.2 Output type.

2.1.1.3. Text

The input string can be some text. This text can be provided inline, using a single-line text with variables or a script expression.

2 properties are requested when the input is a private data:

- **Text**
  The text to be used for conversion (mandatory)
- **Add incoming job extension on output**
  The incoming job extension can be added or omitted on output (when the output is a renamed job) – default: 'Yes'

When the input string is a text, the output can be:

- **Rename incoming job**
- **Private data**

By default, this property is set to 'Rename incoming job'.

The output settings are detailed in 2.1.2 Output type.

2.1.1.4. Dataset

The input can also be one of the datasets of the job. In this case, the conversion only allows to clean the dataset or create a new dataset from the cleaned original one. The dataset name can be provided inline, using a single-line text with variables or a script expression.

When cleaning a dataset, the incoming job name is not touched and will be preserved on output.

1 property is requested when the input is a dataset:

- **Dataset name**
  The name of the dataset to be used for conversion (mandatory)

Dataset model is automatically detected and can be XML, XMP, JDF, JSON or OPAQUE.

Keep in mind that StringWizard can only handle text-based files, such as XML, XMP, JDF, JSON or TXT. Attempting to use it on binary files (attached as an opaque dataset) may result in unexpected outcomes, such as damaged and illegible files.

It is strongly recommended to only use 'Convert diacritics and ligatures', 'Convert trademarks' and 'Custom conversion' (if properly setup) and avoid 'Convert case, 'Convert space and 'Convert reserved characters' when using XML, XMP, JDF or JSON datasets to avoid destroying file structure, keys or tags.

2.1.2. Output type for incoming job, private data and text

There are 2 types of output that can be provided by StringWizard:

- **Rename incoming job**
- **Private data**

By default, this property is set to 'Rename incoming job'.

2.1.2.1. Rename incoming job

If the selected output is 'Rename incoming job', the incoming job is going to be renamed with the processed string from the input.

2.1.2.2. Private data

If the selected output is 'Private data', another property is requested:

- **Key**
  The private data key to be used (mandatory) – default: 'aazappCleanedString'

2.1.3. Output type for dataset

When cleaning a dataset, the output is a dataset attached to the job.
There are 2 types of output that can be provided by StringWizard:

- **Replace original datatset**
- **New dataset**

By default, this property is set to 'Replace original dataset'.

2.1.3.1. Replace original dataset

If the selected output is 'Replace original dataset', the submitted dataset is cleaned by StringWizard and the cleaned version of the dataset replaces the original dataset.

If you replace the original dataset by its cleaned version, there is no way to retrieve the original dataset. Be careful of your settings before using this option. If you are not sure of your settings, please use 'New dataset' instead.

2.1.3.2. New dataset

If the selected output is 'New dataset, a new dataset is created and attached to the job. The model and extension of the original dataset is used on output. When 'New dataset' is chosen, another property is requested:

- **Dataset name**
  The name of the new (cleaned) dataset (mandatory)

2.1.4. Use case examples

Here is how to set up input and output to serve the following use cases:

- **Rename the incoming job using the job's name**
  Input: 'Incoming jobname' / Output: 'Rename incoming job' – choose between jobname proper and complete jobname

- **Rename the incoming job using a (cleaned) private data**
  Input: 'Private data' / Output: 'Rename incoming job' – don't forget to add incoming extension if you want to keep it on the outgoing job
- **Store the cleaned jobname as a private data**
  Input: 'Incoming jobname' / Output: 'Private data' – the cleaned jobname will be added as a private data to the job, but the job keeps its original name
- **Clean a private data**
  Input: 'Private data' / Output: 'Private data' – use the same key to update the private data, another name to create a new cleaned private data
- **Clean a dataset**
  Input: 'Dataset' / Output: 'Replace original dataset' – clean the dataset and keep its name, so the same reference can be used throughout the flow
- **Create a cleaned dataset from an existing one**
  Input: 'Dataset' / Output: 'New dataset' – create a new dataset and keep the original dataset, which is the safest option but require to point to a different dataset within the flow

2.2. Convert diacritics and ligatures

StringWizard can convert diacritics (see 3.1 for definition) and ligatures (see 3.2 for definition).

If set to 'Yes', diacritics will be converted to their equivalent without alteration:

- éléphant ➜ elephant

Ligatures will also be replaced by their equivalent:

- œuf ➜ oeuf

By default, this property is set to 'Yes'.

2.3. Convert trademarks

StringWizard can convert trademarks (see 3.3 for definition).

If set to 'Yes', the trademark will be replaced by a sequence of characters:

- © ➜ (C)
- ® ➜ (R)
- ™ ➜ (TM)

By default, this property is set to 'No'.

2.4. Convert case

The case of the processed string can be changed. The different values for this property are:

- **Keep**
  This is the String to Modify ➜ This is the String to Modify (no change)
- **lowercase**
  This is the String to Modify ➜ this is the string to modify
- **UPPERCASE**
  This is the String to Modify ➜ THIS IS THE STRING TO MODIFY

By default, this property is set to 'lowercase'.

2.5. Convert space

Spaces in the processed string can be kept, removed or replaced. The different values for this property are:

- **Keep**
  This is the String to Modify ➜ This is the String to Modify (no change)
- **Remove**
  This is the String to Modify ➜ ThisistheStringtoModify
- **Dash**
  This is the String to Modify ➜ This-is-the-String-to-Modify
- **Underscore**
  This is the String to Modify ➜ This_is_the_String_to_Modify

By default, this property is set to 'Underscore'.

2.6. Convert reserved characters

Reserved characters (see 3.4 for list of reserved characters) in the processed string can be kept, removed or replaced. The different values for this property are:

- **Keep**
  Don't use this! ➜ Don't use this! (no change)
- **Remove**
  Don't use this! ➜ Dont use this
- **Dash**
  Don't use this! ➜ Don-t use this-
- **Underscore**
  Don't use this! ➜ Don_t use this_

By default, this property is set to 'Underscore'.

2.7. Custom conversion

On top of preset conversions (see from 2.2 to 2.6), there is a possibility to create your own pairs of conversion.

If set to 'Yes', you will be asked to enter the key pairs for custom conversion. This property can be provided using a multi-line text with variables or a script expression:

- **Conversion table**
  The table with all key pairs, one line per conversion scheme, key pairs separated by <=> between input and output values.

This property is mandatory.

2.7.1.1. Key pairs definition in the conversion table

You can define several kinds of conversion.

On the left side (input), you can use one character, a combined character (one of the methods for diacritics), several characters, regular expression or text with variables.

On the right side (output), you can use one character, a combined character (one of the methods for diacritics), several characters or text with variables.

To improve the layout of the table (easier to read), you can use commenting lines.
These lines start with // and will be ignored by the conversion. Empty lines are also ignored.

For example:

```
// Replace '/' by '\'
/ <=> \

// Next comment
```

Here are some use examples of combination:

- **One character or combined characters <=> one character, combined characters or text with variables**
  â <=> a
  Ltd <=> Limited
  currentDate <=> [Switch.Date:Format="yyyy-MM-dd",TimeZone="System"]
- **Regular expression <=> one character or combined characters**
  (p|P)df <=> PDF
  die-cut|dieline|découpe|Stanz <=> diecut

The possibilities are endless and can cover a lot of use cases.

For example:

- **Reserved characters**
  You don't want to clean all reserved characters from the default list, but only some of them, you can just create one custom conversion for these ones.
- **Missing diacritic**
  You just must create your own key pair for this specific diacritic.
  Please do get in touch with us and tell us what is missing, so we can add it to the list!
- **Replace abbreviation by full name and vice-versa**
  Create your own list of conversions for these (e.g. Inc <=> Incorporated)
- **Make an XML compatible between different systems**
  System 1 is producing a tag in a certain form, but system 2 needs to get a different value. In that case, you just use a key pair, like: value_from_system_1 <=> value_for_system_2

If you have a list of conversions (more than one line), they are executed from top to bottom, so the order is important! Also remember that the custom conversion is executed after all conversions (except truncation), so be sure of your settings, as you may have interactions between them.

2.8. Truncate string

The string to be processed can be truncated. When truncation is set to 'Yes', the 'Keep' property is displayed and must be filled in. Entering a positive number will truncate the string from the beginning, a negative number from the end of the string.

- **7**
  "Keep some characters" ➜ "Keep so"
- **-6**
  "Keep some characters" ➜ "acters"

By default, this property is set to 'No'.

Note that:

- One space count as one character
- If the number entered is greater than the number of characters of the string, the string will not be truncated!
- Keep in mind that converting ligatures, trademarks (and some diacritics) will increase the number of characters (e.g. œ ➜ oe). This is the reason why the truncation is done at the end of the process to avoid getting a wrong number of requested characters

3. App private data

3.1. Store processing information

StringWizard stores all information about the conversion process in private data.

These private data can be used for later processing (such as renaming back to original) or audit trail:

- **stringWizard.Info.Input.Original**
  The original filename with extension (see 2.1.1 Input type)
- **stringWizard.Info.Input.Value**
  The original string sent to StringWizard, or dataset model if input type is dataset (see 2.1.1 Input type)
- **stringWizard.Info.Input.Type**
  The type of input for the conversion (see 2.1.1 Input type)
- **stringWizard.Info.Input.PrivateDataKey**
  The private data key if input type is private data (see 2.1.1.2 Private data)
- **stringWizard.Info.Input.DatasetName**
  The name of the input dataset if input type is dataset (see 2.1.1.4 Dataset)
- **stringWizard.Info.Input.DatasetModel**
  The model of the input dataset if input type is dataset (see 2.1.1.4 Dataset)
- **stringWizard.Info.Input.DatasetExtension**
  The extension of the input dataset if input type is dataset (see 2.1.1.4 Dataset)
- **stringWizard.Info.Clean.Diacritics**
  Boolean key for the diacritics conversion (see 2.2 Convert diacritics and ligatures)
- **stringWizard.Info.Clean.Trademarks**
  Boolean key for the trademarks conversion (see 2.3 Convert trademarks)
- **stringWizard.Info.Clean.Case**
  Boolean key for the case conversion (see 2.4 Convert case)
- **stringWizard.Info.Clean.Space**
  Boolean key for the space conversion (see 2.5 Convert space)
- **stringWizard.Info.Clean.Reserved**
  Boolean key for the reserved characters conversion (see 2.6 Convert reserved characters)
- **stringWizard.Info.Clean.Custom**
  Boolean key for the custom conversion (see 2.7 Custom conversion)
- **stringWizard.Info.Clean.Custom.Table**
  Table used for custom conversion (see 2.7 Custom conversion)
- **stringWizard.Info.Clean.AddExtension**
  Boolean key for adding job extension if input type is private data or text (see 2.1.1 Input type)
- **stringWizard.Info.Clean.Truncate**
  Boolean key for the truncation (see 2.8 Truncate string)
- **stringWizard.Info.Clean.Truncate.Keep**
  The number of characters to be kept (see 2.8 Truncate string)
- **stringWizard.Info.Output.Value**
  The resulting string after conversion, or dataset model if output type is dataset (see 2.1.2 & 2.1.3 Output type)
- **stringWizard.Info.Output.Type**
  The type of output for the conversion (see 2.1.2 & 2.1.3 Output type)
- **stringWizard.Info.Output.PrivateDataKey**
  The private data key if output type is private data (see 2.1.2 Output type for private data)
- **stringWizard.Info.Output.DatasetName**
  The name of the output dataset if output type is dataset (see 2.1.3 Output type for dataset)
- **stringWizard.Info.Output.DatasetModel**
  The model of the output dataset if output type is dataset (see 2.1.3 Output type for dataset)
- **stringWizard.Info.Output.DatasetExtension**
  The extension of the output dataset if output type is dataset (see 2.1.3 Output type for dataset)

4. Glossary

4.1. Diacritic

Diacritic is a glyph added to a letter that somehow alters its pronunciation, and which appears above, below, within or between letters — French uses five diacritics: the grave, acute, and circumflex accents, as well as the cedilla and dieresis.

4.2. Ligature

Ligature is a link between two printed letters

4.3. Trademark

Trademark us a symbol, word, or words legally registered or established by use as representing a company or product.

StringWizard can convert the following symbols:

©	®	™

4.4. Reserved character

Reserved character is a character that is not allowed and/or not safe for web or OS filenames.

Currently, the reserved characters captured by StringWizard are:

,	;	:	.	!	?	*	#	@	%	=	+	'	’	"	`	^	~	&	$	<	>	[	]	{	}	|	\	/

5. More information

5.1. About us

This app was created by aazapp. You can find more information about us here: https://aaz.app. We're geeks about topics such as print, PDF, prepress, preflight, automation, and a good deal more. We like to share our expertise with others, sometimes by doing paid professional services projects and other times by bundling our knowledge into (often free) Switch apps.

When we build an app and lovingly craft the code, branding, documentation, and everything else that goes into it, it's because we like it and think it will be helpful for other people as well—people like you.

5.1.1. Is it?

Rating our app on the Enfocus app store is quick and easy, but it makes us very happy. It feels like a reward for the time we have invested in the app. So please visit the app store (https://www.enfocus.com/en/appstore/overview) and take a second to rate it.

5.1.2. Found a problem?

We don't know your Switch flows, so we can't guarantee that our apps will play nicely with them (however much time we invest in testing before releasing them). Please don't hesitate to reach out to us and ask questions, give feedback, and make suggestions about the apps you're using or about other things that would be useful for you in Switch.

Just send us an email at hello@aaz.app. We'll be with you... ASAP (obviously).

5.2. Sushi

Oh... one more thing!

If you're using our free apps and they are valuable to you, please consider donating a small amount to us. We (the geeks behind aazapp) have a tradition of having sushi together in different cities worldwide, and some pocket money to do so is always appreciated!