

Templates with Scripts

Angelika Zerfaß

zaac



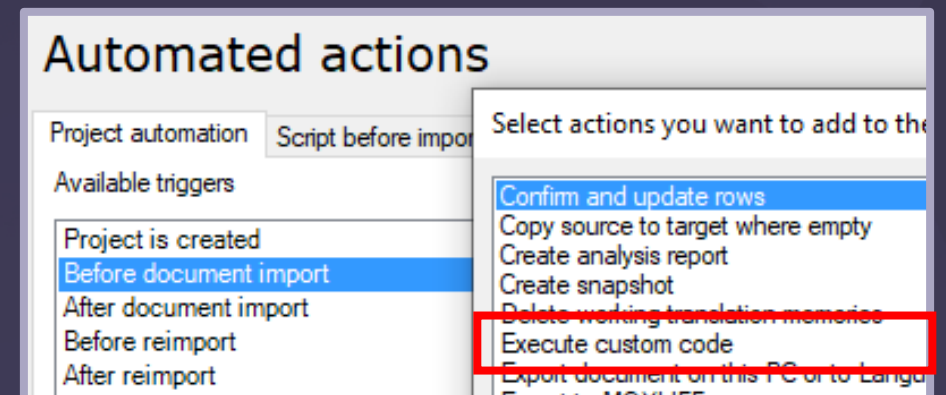
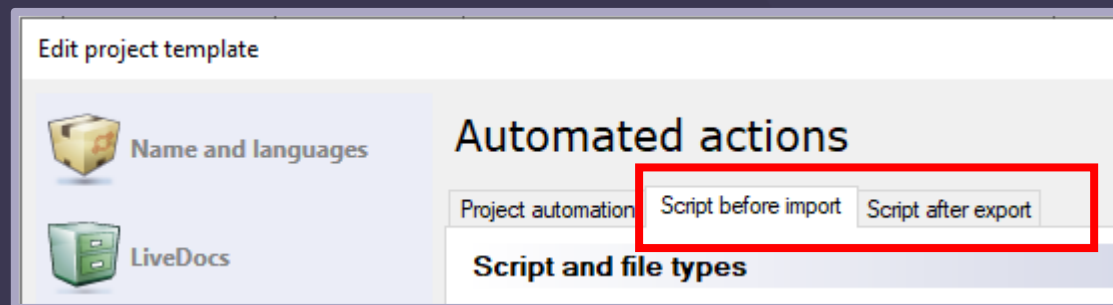
2024

Templates with Scripts

In addition to the existing automated actions, templates also allow you to run a script in different situations (trigger points) in a project.

file processing before import
file processing after export

XLIFF manipulation
during the project



Before Import

Files might need preparation steps to hide text from translation or to adapt the structure of the file for correct processing in memoQ.

- Macros to run on Word files to apply hidden formatting
- Structure changes to XML files to make them multilingual
- Changing line break characters in the text inside an XML file from (\n) into tags <break/> for better segmentation


Before Import - Macros in Word

Example:

- Hiding all text that is not marked with a yellow highlight color

Original document

Translate all text with yellow highlight

ID	Item	Image
123-456	Left Rear Window Shade Button	
345-678	Taster für Seitenrollo hinten rechts	
789-345	Schalter hinten für Heckrollo	
678-124	Rear-Right Outside Door Handle Central Locking Button	

In memoQ

(after preparation with a macro to apply hidden text to everything that does NOT have a yellow highlight)

View pane

	Taster für Seitenrollo hinten rechts	
	Schalter hinten für Heckrollo	

Before Import - Macros in Word

Example:

- Adding a copy of each paragraph so that after translation there is a paragraph in the source language followed by the translated paragraph.

This-is-a-segment.¶

This-is-another-segment.·And-another-one.¶

Some-more-segments-here.·And-still-more-text.·¶

This-is-a-segment.¶

This-is-a-segment.¶

This-is-another-segment.·And-another-one.¶

This-is-another-segment.·And-another-one.¶

Some-more-segments-here.·And-still-more-text.·¶

Some-more-segments-here.·And-still-more-text.·¶

- copy paragraphs
- apply hidden formatting
to source

This-is-a-segment.¶

Dies-ist-ein-Segment.¶

This-is-another-segment.·And-another-one.¶

Dies-ist-ein-weiteres-Segment.·Und-ein-weiteres.¶

Some-more-segments-here.·And-still-more-text.·¶

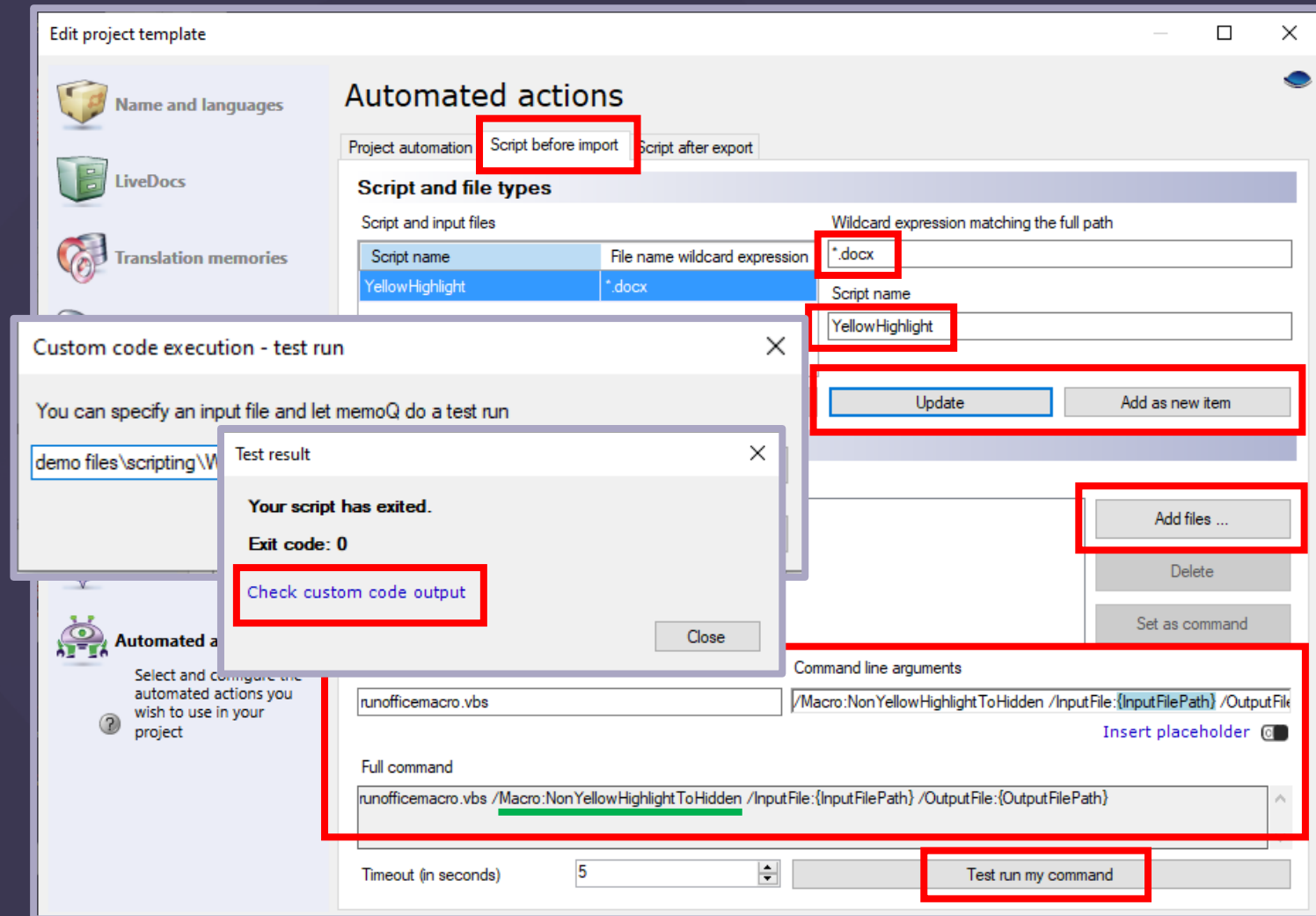
Mehr-Segmente.·Und-noch-mehr-Text.¶

Only visible text
gets translated

Setting up the template

Script before import tab

- Add file extension
- Enter a name
- Add the scripting file
- Add the command that includes the macro name
- Add/Update item
- Test your automation
 - The blue text opens a temp folder with the result of the script



2024

After Export – Macros in Word

Set up the corresponding options to run a macro on the Word files **after export** to unhide the hidden text again.

It works the same way as the Scripting before Import, just change the name of the macro to be used.

Full command

```
runofficemacro.vbs /Macro:TurnHiddenOff /InputFile:{InputFilePath} /OutputFile:{OutputFilePath}
```

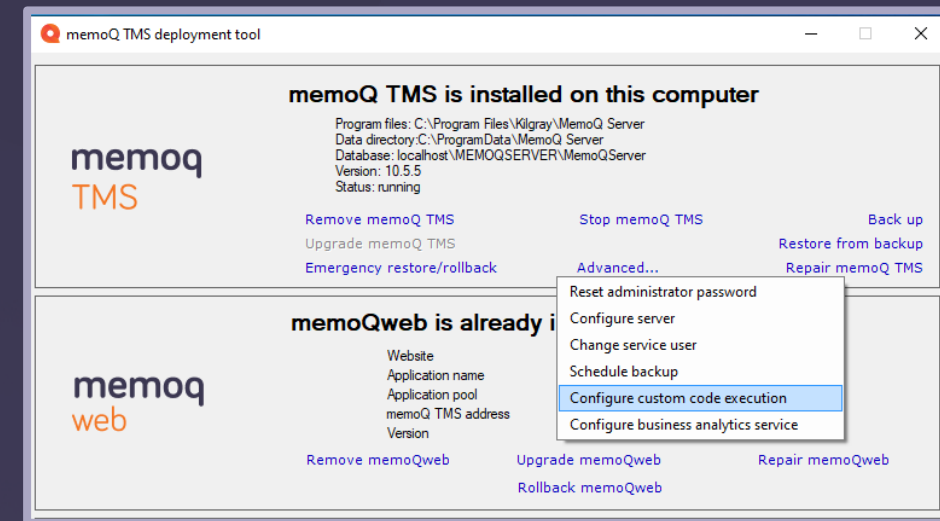
Scripting in server projects

1. To use macros memoQ server projects, Office has to be installed **on the machine that runs memoQ server!**
2. The server needs to be **allowed to run scripts.**

- Change settings in the deployment tool on the memoQ server machine.

- See the online help for further explanation:

<https://docs.memoq.com/current/en/Places/edit-template-import-export-scripting.html>



Before Import - Structure Changes

An XML file needs to be multilingual after translation.

- memoQ cannot add new XML structure tags into an existing file, but a script can prepare the file so that the necessary structure for the translations exist.

```
<strings>
<msg ID="001">
<en>This is the message</en>
</msg>
</strings>
```

```
<strings>
  <msg ID="001">
    <en>This is the message</en>
    <it/>
    <de/>
    <es/>
    <fr/>
  </msg>
</strings>
```

Before Import – Segmentation Markers

```
<?xml version="1.0" encoding="UTF-8"?>
<body>
  <content>Here is some text. \nWhenever there is a linebreak, the notation backslash
  and "n" is used. \nmemoQ should segment at the linebreak marker.
</content>
```

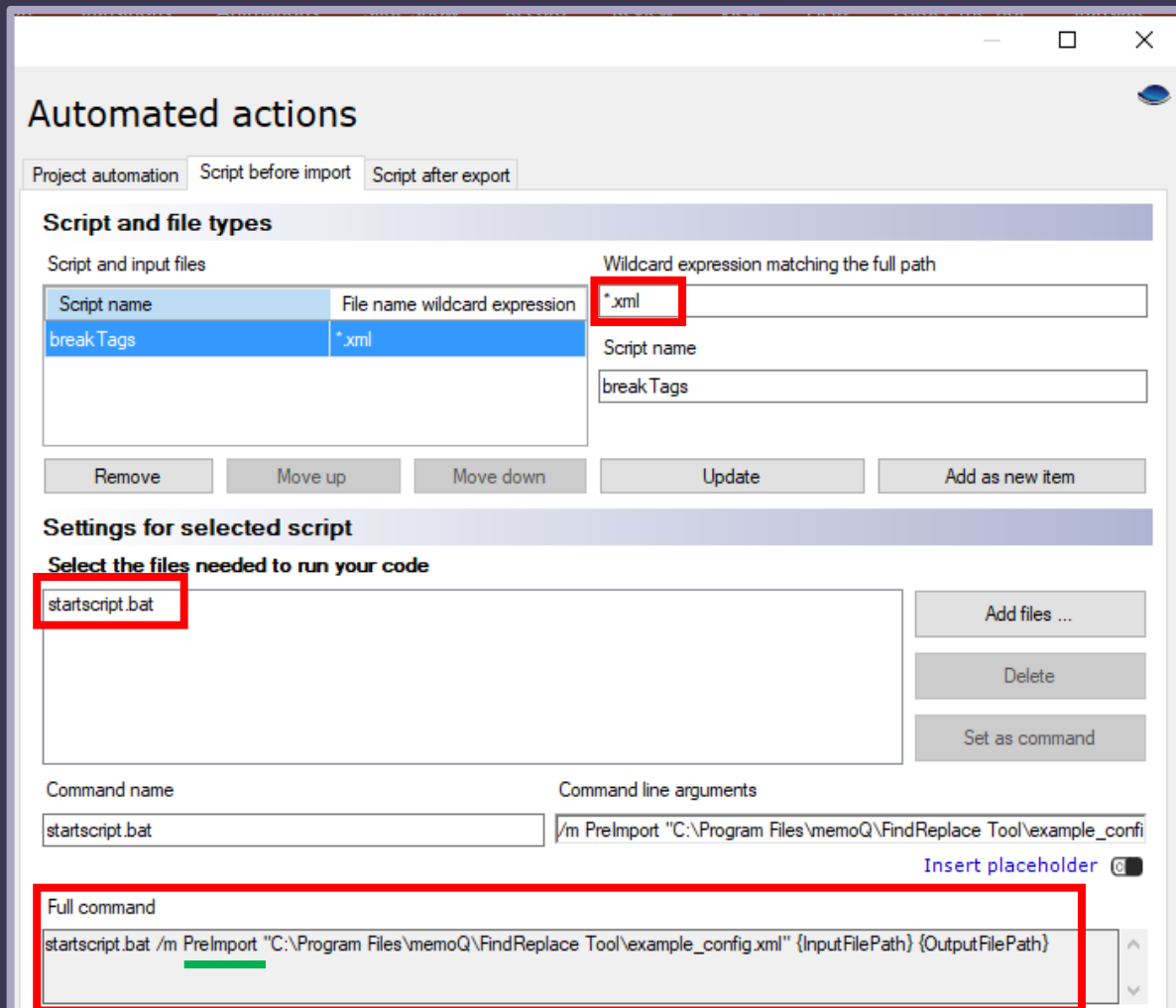
Changing the import file so that memoQ recognizes segmentation points.

- Here an XML file that contains text from another system which uses codes like `\n` for line breaks.
- A script can change that into a code that memoQ can interpret more easily as a segment break in XML, like `<break/>`.

REGEX – Find and Replace

- Simple find&replace actions like these can be created with the **FindAndReplaceTool** that comes with memoQ.
- It is installed at: C:\Program Files\memoQ\FindReplace Tool
 - (This is also the place where the configuration files with the find&replace actions need to be located)
- See also: <https://docs.memoq.com/current/en/Places/edit-template-find-and-replace.html>
- The tool can manipulate files before import, files after export **AND** files after import (MQXLIFF).

Segmentation Changes



Example for segmentation change before import and undoing the change after export.

```

--<memoQFAndReplace xsi:noNamespaceSchemaLocation="sample.xsd">
  --<usecase name="PreImport">
    --<searchItem type="text">
      <searchExpression> \n </searchExpression>
      <replaceExpression> <break/> </replaceExpression>
    </searchItem>
  </usecase>
  
```

```

--<usecase name="PostExport">
  --<searchItem type="text">
    <searchExpression> <break/> </searchExpression>
    <replaceExpression> \n </replaceExpression>
  </searchItem>
</usecase>
  
```

Manipulating the XLIFF files

In some instances you might want to change things in the XLIFF files inside the project. For example:

- changing/deleting things done by pre-translation
 - adjusting MT suggestions
 - deleting pseudo-translation
- setting a segment to locked
- changing the segment status

Manipulating the XLIFF files

The FindAndReplace Tool of memoQ will do the following:

- At the desired trigger point, run the "Execute Custom Code" action.
- The file in the project is exported as MQXLIFF to a temp folder and processed with a regular expression find&replace action.
- Then the MQXLIFF file is imported back to the project and updates the document there.

FindAndReplace Tool

What you need:

- Script file that initiates the custom code execution (comes with the FindAndReplace Tool, startscript.bat).
- Configuration XML that contains the find & replace action
 - Example config file comes with the FindAndReplace Tool
 - Define regular find&replace or REGEX action

Running Custom Code

- At the trigger points in the automation section, select "Execute custom code".
- On a memoQ sever the server needs to allow execution of the code!

Automated actions

Project automation | Script before import | Script after export

Available triggers

- Project is created
- Before document import
- After document import**
- Before reimport
- After reimport
- After bilingual update
- Document is removed
- After project wrap-up

Actions added to the selected trigger

Action	Scope
Pre-translate	Documents
Execute custom code	Documents

Specify custom code details

Command name and arguments

Select the files needed to run your code

startscript.bat

Add files ...

Delete

Set as command

Command name: startscript.bat

Command line arguments: /m PostImport "C:\Program Files\memoQ\FindReplace Tool\example22_

Insert placeholder

Full command: startscript.bat /m PostImport "C:\Program Files\memoQ\FindReplace Tool\example22_config.xml" {InputFilePath} {OutputFilePath}

Timeout (in seconds): 5

Export documents as MQXLIFF for script to process

Update documents in project from script's output

The script's output file must be named <input-file-name>_out. The file extension needs to stay the same.

Test your command

You can specify an input file and let memoQ do a test run

Select input file

Run

FindAndReplace Tool – Configuration File

```
--<memoQFAndReplace xsi:noNamespaceSchemaLocation="sample.xsd">
```

```
--<usecase name="PreImport">
  --<searchItem type="text">
    <searchExpression> \n </searchExpression>
    <replaceExpression> <break/> </replaceExpression>
  </searchItem>
</usecase>
```

Changing line break characters into break tags before import

```
--<usecase name="PostImport">
  --<searchItem type="regex">
    --<searchExpression>
      (?s)(?<TransunitStart><trans-unit [^>]*id="\d+" )(?!<TransunitEnd>[>]*>\s*<source[^>]*>)(?<sourceText>\d+)(?<midtext></source>\s*<target[^>]*)(/>|></target>)
    </searchExpression>
    --<replaceExpression>
      ${TransunitStart}translate="no"mq:locked="locked" ${TransunitEnd}${sourceText}${midtext}>${sourceText}</target>
    </replaceExpression>
  </searchItem>
</usecase>
```

Adding the information "locked" to digit-only segments

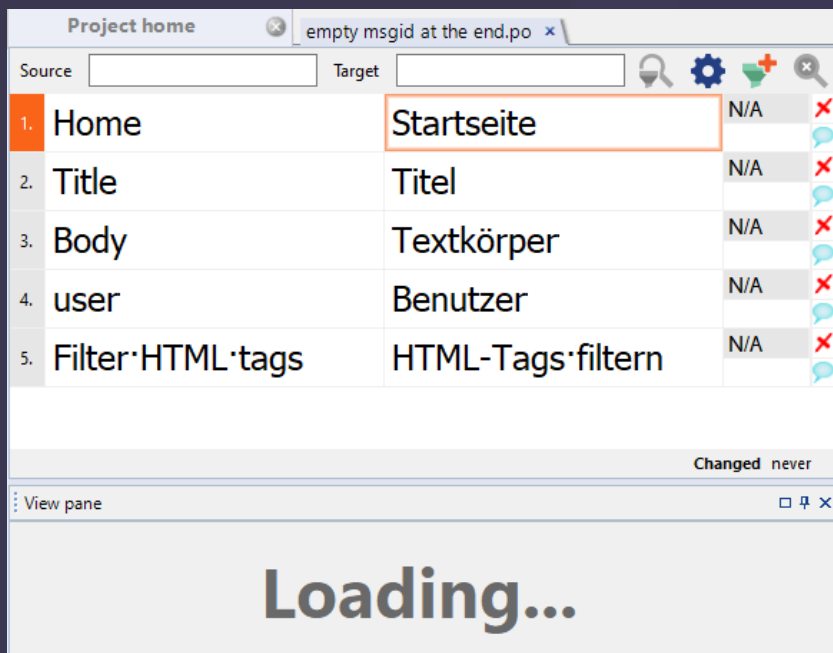
```
--<usecase name="PostExport">
  --<searchItem type="text">
    <searchExpression> <break/> </searchExpression>
    <replaceExpression> \n </replaceExpression>
  </searchItem>
</usecase>
</memoQFAndReplace>
```

Changing break tags back to line break characters after export

PO file (with empty element)

PO files with an empty element (spaces only) prevent memoQ from creating a preview. Changing the spaces before import to some other characters allows the preview creation. Script to undo the changes after export.

Without preparation



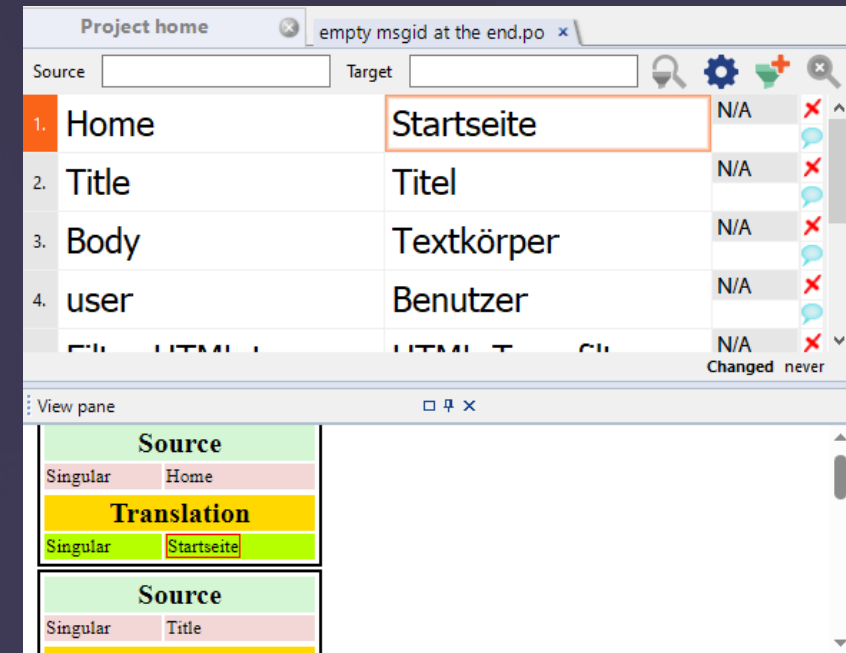
Source	Target		
1. Home	Startseite	N/A	✘
2. Title	Titel	N/A	✘
3. Body	Textkörper	N/A	✘
4. user	Benutzer	N/A	✘
5. Filter·HTML·tags	HTML-Tags·filtern	N/A	✘

Changed never

View pane

Loading...

With preparation



Source	Target		
1. Home	Startseite	N/A	✘
2. Title	Titel	N/A	✘
3. Body	Textkörper	N/A	✘
4. user	Benutzer	N/A	✘
5. Filter·HTML·tags	HTML-Tags·filtern	N/A	✘

Changed never

View pane

Source	
Singular	Home
Translation	
Singular	Startseite

Source	
Singular	Title

More...

- Changing quote marks " to « and » after pre-translation with MT (DeepL in this case).
 - Added a status change for these segments (status = Rejected) to alert the users that this segment has been changed.
- Deleting the pseudo translation (after import) to keep the file in the project for further processing .
 - No need to re-import a clean copy of the file in case the pseudo translation did not find any issues.
- Extracting segments with comments from the XLIFF file via a Python script.



Your Ideas...

zerfass@zaac.de