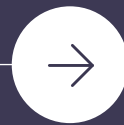


memoQ views: speeding up the snail-paced handling of complex XML files



Lana Taratukhina

Alpha CRC





The challenges

- Multiple products within one file (varies from 5 to 8 products)
- 36 target languages
- Different product TMs to be used, i.e. one project per product
- Separate quotes – one per product
- All translated products to be delivered within one file



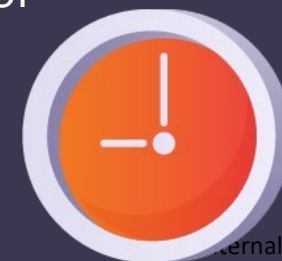


If not the views – then what?

!

Embracing extra time and hassle:

- ❑ Each product would have to be imported with a custom RegexpText filter, with a separate filter per product
- ❑ Only after the first product is translated for all languages, it would be possible to export the file and import the second product into another project, etc...
- ❑ ...Or we'd have to write a complex script that would split the file according to products and then glue it back together
- ❑ The script would have to be updated regularly, as the names of products often change



The views approach



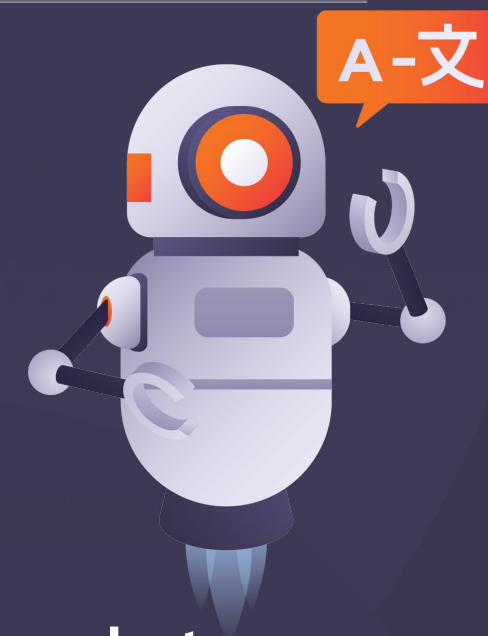
Creating online projects – one per product



Creating the local Master project – one for all products



Importing the received XML file into the Master project – ALL products



```

<data name="CALENDARFILE_INVALID" xml:space="preserve">CRIF
  <value>Calendar file is invalid.</value>CRIF
  <comment>MUI</comment>CRIF
</data>CRIF
<data name="SORTS_JOB_BY" xml:space="preserve">CRIF
  <value>Sort jobs by</value>CRIF
  <comment>MUI</comment>CRIF
</data>CRIF
<data name="SORT_BY_NAME_ASCENDING" xml:space="preserve">CRIF
  <value>Name (A...Z)</value>CRIF
  <comment>MUI</comment>CRIF
</data>CRIF
<data name="SORT_BY_NAME_DESCENDING" xml:space="preserve">CRIF
  <value>Name (Z...A)</value>CRIF
  <comment>MUI</comment>CRIF
</data>CRIF
<data name="SORT_BY_DATE_ASCENDING" xml:space="preserve">CRIF
  <value>Date created -- Most recent to least recent</value>CRIF
  <comment>NGHMIJ</comment>CRIF
</data>CRIF
<data name="SORT_BY_DATE_DESCENDING" xml:space="preserve">CRIF
  <value>Date created -- Least recent to most recent</value>CRIF
  <comment>NGHMIJ</comment>CRIF
</data>CRIF
<data name="REQUEST_TIMEOUT" xml:space="preserve">CRIF
  <value>Request timeout, did not receive firmware file request from controller.</value>CRIF
  <comment>MUI</comment>CRIF
</data>CRIF
<data name="FAILED_TO_TRANSFER_LANGUAGE_FILES" xml:space="preserve">CRIF
  <value>Failed to transfer Language files</value>CRIF
  <comment>TTO</comment>CRIF
</data>CRIF
<data name="INVALID_XML_FILE" xml:space="preserve">CRIF
  <value>Coundn't parse file invalid xml file</value>CRIF
  <comment>TTO</comment>CRIF
</data>CRIF

```

Multi-product XML files received from the client

The XML filter configuration for Master project

Filter: Cascading filter

Filter configuration: MARK - resx w comments and ID

Add cascading filter... | Remove

XML filter > Regex tagger

Encoding and reference files | General | **Tags and attributes** | Entities | Subtitles

Handled tags	
Name	Info
comment	Str;NT;Inh;Com
data	Str;Pres
resheader	Str;NT;Inh
root	Str;Inh
value	Str;Pres
xsd:attribute	Str;Inh
xsd:choice	Str;Inh
xsd:complexType	Str;Inh
xsd:element	Str;Inh
xsd:import	Str;Inh

Product name as a comment

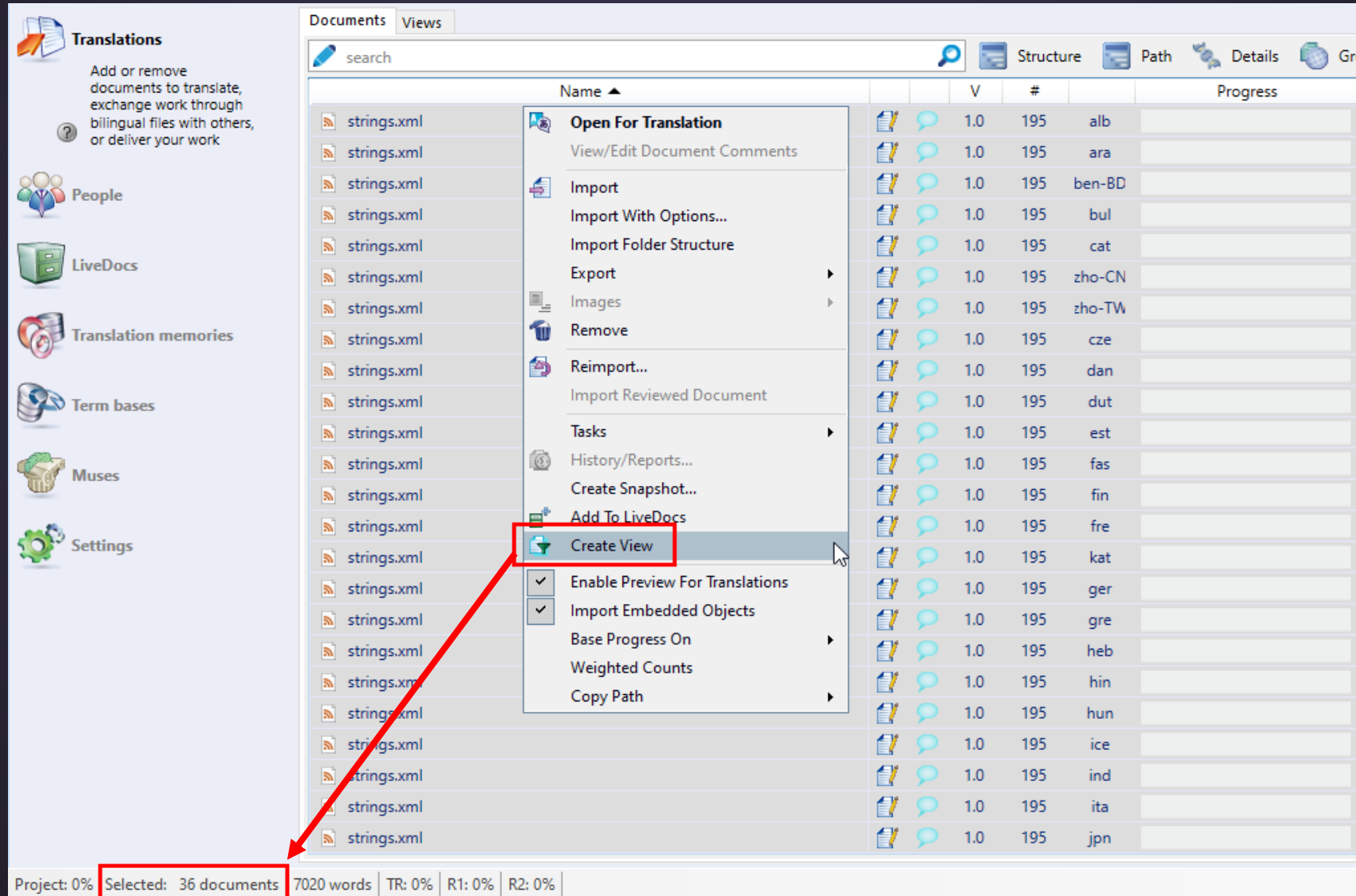
Text to translate

Tag settings

- Inline
- Not translated
- Required: Normal
- Whitespace handling: Inherit
- Tag content is context ID for siblings
- Tag content is comment for siblings
- Siblings as: Only following

Clear list | Populate

Creating 36 views for one product in one click



The screenshot shows the memoQ interface with a list of 36 documents, all named 'strings.xml'. A context menu is open over one of the documents, and the 'Create View' option is highlighted. A red arrow points from this option to the status bar at the bottom, which shows 'Selected: 36 documents'.

Name	V	#	Progress
strings.xml	1.0	195	alb
strings.xml	1.0	195	ara
strings.xml	1.0	195	ben-BD
strings.xml	1.0	195	bul
strings.xml	1.0	195	cat
strings.xml	1.0	195	zho-CN
strings.xml	1.0	195	zho-TW
strings.xml	1.0	195	cze
strings.xml	1.0	195	dan
strings.xml	1.0	195	dut
strings.xml	1.0	195	est
strings.xml	1.0	195	fas
strings.xml	1.0	195	fin
strings.xml	1.0	195	fre
strings.xml	1.0	195	kat
strings.xml	1.0	195	ger
strings.xml	1.0	195	gre
strings.xml	1.0	195	heb
strings.xml	1.0	195	hin
strings.xml	1.0	195	hun
strings.xml	1.0	195	ice
strings.xml	1.0	195	ind
strings.xml	1.0	195	ita
strings.xml	1.0	195	jpn

Project: 0% Selected: 36 documents 7020 words | TR: 0% | R1: 0% | R2: 0%

Views creation



memoQ will batch-create 36 views based on the comment that contain only one product



The process to be repeated for all products

Create view (filtering and sorting)
✕

Name of the view

Filter by frequency

Minimum frequency

Keep duplicates

Segment range

Only segments from row If your view is based on a single document, you can select a range of segments to be included

to row

Filter by text

Source content Case sensitive

Add

Remove

Enter words or "longer expressions" that must ALL be present in the segment. All segments that meet ANY of the rows are included.

Filter by segment status

Common filters
Status
Conflicts and changes
Comment and tags

Comment contains

Row has active language quality error

Source has memoQ {tag}

Target has memoQ {tag}

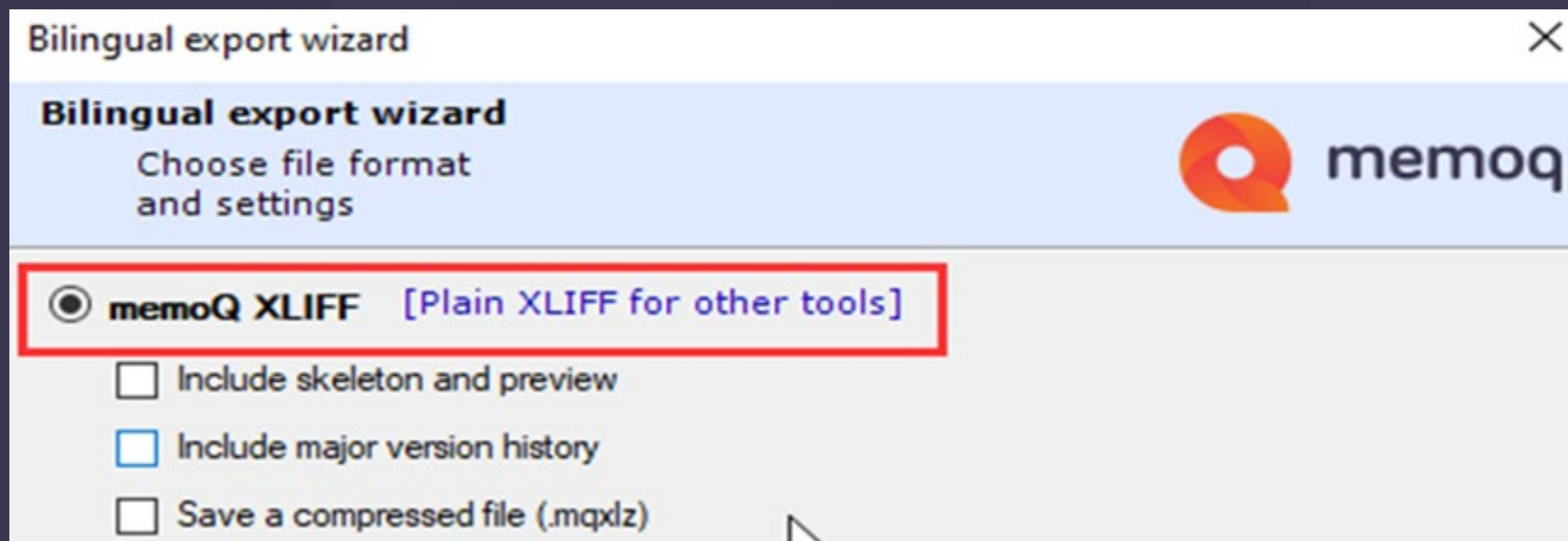
Source has inline tag

Target has inline tag

Views export



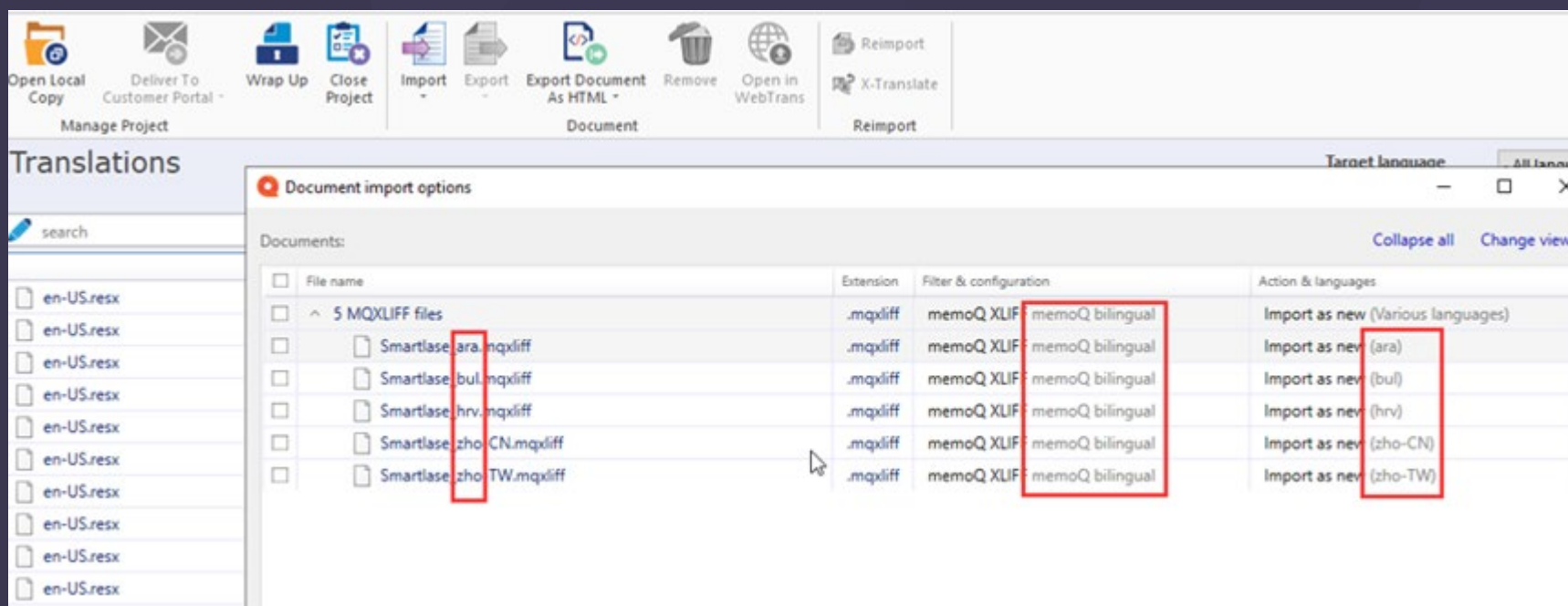
The PM will export bilingual views to be used in online projects



Views import into the online project



- The bilingual views = new source files, one file per product/language
- Easy to just drag and drop, no need to even select the filter



The screenshot shows the 'Document import options' dialog box in the memoQ software. The dialog lists 5 MQXLIFF files. The 'Filter & configuration' column shows 'memoQ XLIFF' and 'memoQ bilingual' for each file. The 'Action & languages' column shows 'Import as new' followed by language codes in parentheses: '(Various languages)', '(ara)', '(bul)', '(hrv)', '(zho-CN)', and '(zho-TW)'. Red boxes highlight the 'memoQ bilingual' filter and the language codes in the 'Action & languages' column.

File name	Extension	Filter & configuration	Action & languages
^ 5 MQXLIFF files	.mqxliff	memoQ XLIFF memoQ bilingual	Import as new (Various languages)
Smartlase_ara.mqxliff	.mqxliff	memoQ XLIFF memoQ bilingual	Import as new (ara)
Smartlase_bul.mqxliff	.mqxliff	memoQ XLIFF memoQ bilingual	Import as new (bul)
Smartlase_hrv.mqxliff	.mqxliff	memoQ XLIFF memoQ bilingual	Import as new (hrv)
Smartlase_zho-CN.mqxliff	.mqxliff	memoQ XLIFF memoQ bilingual	Import as new (zho-CN)
Smartlase_zho-TW.mqxliff	.mqxliff	memoQ XLIFF memoQ bilingual	Import as new (zho-TW)

Updating the Master project





After the product views are translated, the PM will update the Master project with translated views



The Master file can be safely updated with all languages simultaneously, one product at the time – without overwriting products

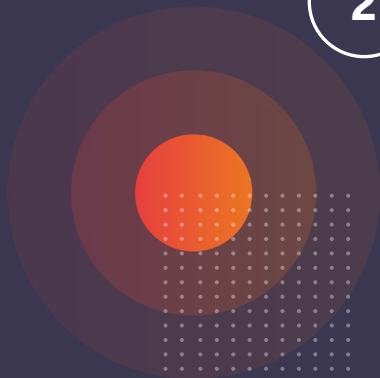
Document import options

Documents: Collapse all

<input type="checkbox"/> File name	Extension	Filter & configuration	Action & languages
<input type="checkbox"/> ^ 2 MQXLIFF files	.mqxliff	memoQ XLIFF memoQ bilingual	Update (bul)
<input type="checkbox"/>  MUI_bul.mqxliff	.mqxliff	memoQ XLIFF memoQ bilingual	Update MUI (bul)
<input type="checkbox"/>  TTO_bul.mqxliff	.mqxliff	memoQ XLIFF memoQ bilingual	Update TTO (bul)

Delivery time!

- 1 Update all the Master files with all views via bilingual update
- 2 Export Master files and deliver all requested products within one Master file





Thank you!

Any questions?

staratukhina@alphacrc.com

