Qucamba Reports

Translations Guide

Doc. Ver. 4.00 Oct 1st, 2025

Qucamba GmbH
Inselstr. 41, 22297 Hamburg
Kapellenstr. 25, 91341 Röttenbach

https://qucamba.com

DISCLAIMER

THIS DOCUMENT OR SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL QUCAMBA GMBH OR ANY OF THEIR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS DOCUMENT OR SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Qucamba GmbH and its contributors assume no responsibility for errors or omissions in the software or documentation available from the qucamba.com web site.
In no event shall Qucamba GmbH and its contributors be liable to you or any third parties for any special, punitive, incidental, indirect or consequential damages of any kind, or any damages whatsoever, including, without limitation, those resulting from loss of use, data or profits, whether or not Qucamba GmbH or its contributors have been advised of the possibility of such damages, and on any theory of liability, arising out of or in connection with the use of this software.
The use of the document or software downloaded through the qucamba.com site is done at your own discretion and risk and with agreement that you will be solely responsible for any damage to your computer system or loss of data that results from such activities. No advice or information, whether oral or written, obtained by you from qucamba.com, its website or its contributors shall create any warranty for the software.

TRADEMARKS
All trademarks referenced from within this document or software which are decorated with or ® are either trademarks or registered trademarks of Microsoft or other companies.

Qucamba is a registered trademark of Qucamba GmbH Germany. The trademark is registered in the EU, CANADA, USA and other countries. All rights are reserved as long as they have not been explicitly granted.

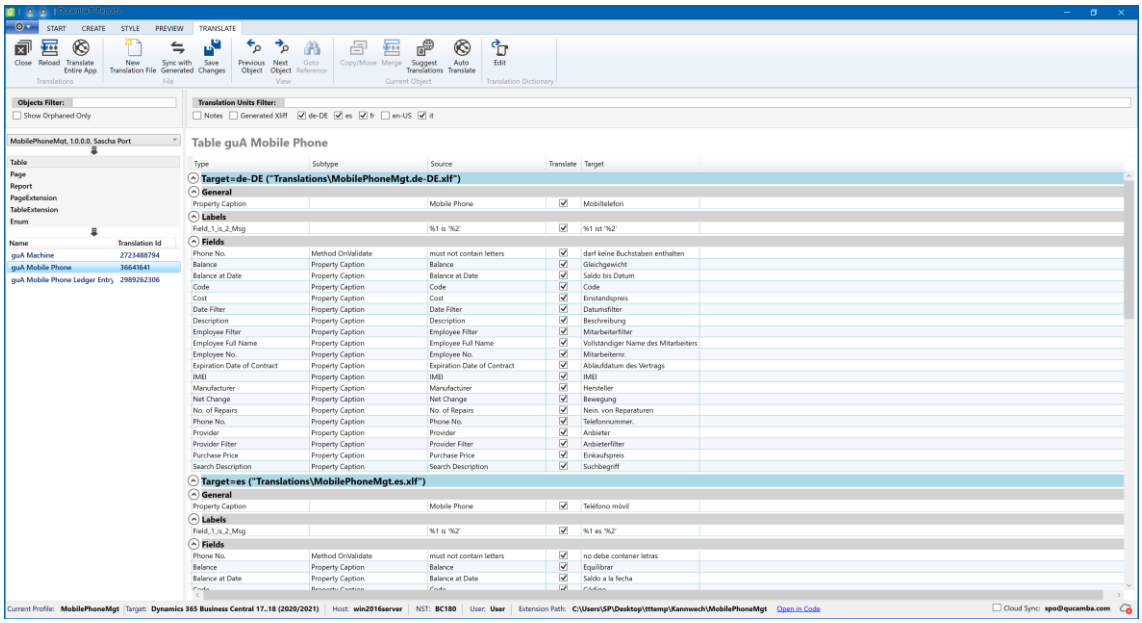
The trademarks Microsoft, Windows, SQL Server, BackOffice are trademarks or registered trademarks of Microsoft Corporation in the USA and/or in other countries. All rights are reserved as long as they have not been explicitly granted.

Table of Content

1 General

Though on first sight, app translations are not associated with developing reports Qucamba Reports comes with a powerful and easy-to-use translation editor. Mainly because reports need to be translated to each app's target language as well, Qucamba Reports contains various tools to help report designers in accomplishing common translation related tasks like copying an entire report including its translations, re-assigning translations after changing symbolic names etc.

Qucamba decided to not restrict the multi-language editor to report objects. Instead, translations of almost all object types and all captions can be translated within Qucamba Reports by using a tabular style editor window with clear-text symbolic names instead of ids. Furthermore, new languages can be added to your app easily.



Various automated translation features make Qucamba Reports one of the most powerful translation tools available today.

2 Understanding Translations in Business Central

A detailed explanation of how to work with translations and language files in Business Central is out of scope of this document. However, a brief understanding of how translations work is mandatory for working with Qucamba Reports TRANSLATE.

This section describes the basic principles of translating Business Central apps.

2.1 Language Codes

Whenever languages need to be referred to, a Locale ID (LCID)[1] is being used.

An example for an LCID is "en-US" which means that the base language is English ("en") and that this language is spoken in the United States ("US"). Another example would be "de-CH" for german language, spoken in Swiss. Also, "de-DE" is a valid LCID which is the same as "de".

2.2 Translation Files

When creating AL objects, a developer should usually write English captions in code. All text literals in AL code refer to the LCID "en-US". Other languages should never appear in AL code – neither in text literals nor in comments. However, some developers still use legacy multilingual properties, such as CaptionML instead of Caption to provide translations to other languages in code files. These properties have been used over years in the old development environment (C/SIDE) and are obsolete today.

Other developers use caption properties and specify translations to other languages in the *Comment* attribute of text literals by using the legacy syntax of the old multilingual properties.

Nowadays, it is very well-known that source code should not contain any text literals at all. Instead, texts and therefore translations as well should be kept separated from code. There are plenty of reasons for this rule which this document is not going to mention. However, in Business Central Microsoft decided to keep the original language (LCID "en-US") text literals in code. This gives a developer a much better understanding of AL code sections that do not have a well describing name. For example, take the following action:

```
action(1000000247)
{
    Caption = 'Combine Shipments';
    […]

    trigger OnAction();
    begin
        // Plenty of code…
```

[1] 2.1.1907 Part 4 Section 7.6.2.39, LCID (Locale ID), https://learn.microsoft.com/en-us/openspecs

```
        end;
    }
```

This action doesn't explain itself by its programmatic name until one can find a describing caption as well. Thus, it became mandatory to specify the "en-US" caption in code.

2.2.1 Preparing an App for Multiple Languages

To include translations to different languages within your app, you add the **TransationFile** feature to the **features** array of your app's **app.json** file as described below.

```
"features": [
  "NoImplicitWith",
  "TranslationFile"
]
```

After saving the **app.json** file and running the build task (VS Code Command "Tasks: Run Build Task" or <Ctrl> + <Shift> + <B>), all text literals in your app's code files are collected and written to a file with the extension ".g.xlf" located in a new "Translations" subfolder of your app's workspace root. Furthermore, each time the app is built again, this file will be recreated. Thus, this file should never be edited, since all changes to this file will get lost on next build.

The ".xlf" file extension stands for "Xliff", which is the file format[2] being used for translating apps and the ".g" stands for "generated".

2.2.2 Adding new languages

To support different languages, one translation file is required for each language. These files are named with the extension {LCID}.xlf with {LCID} as the LCID of the target language e.g. YourAppName.de-DE.xlf would contain all translations for the app to german language.

To create a new language file, you could simply create a copy of the generated file and rename it. However, this is just a naming convention. It doesn't tell Business Central anything about the target language.

Xliff files are basically XML files. To set the target language of a language file, open the file in VS Code and locate the **target-language** attribute in the **file** element right at the beginning of the file. An example Xliff file might look like illustrated below.

```
<?xml version="1.0" encoding="utf-8"?>
<xliff version="1.2" xmlns:xsi=[…] xsi:schemaLocation="[…]" xmlns="[…]>
  <file datatype="xml"
        source-language="en-US"
        target-language="de-DE"
```

[2] XLIFF Version 1.2 OASIS Standard

```
       original="YourApp">
    <body>
      <group id="body">
        […Translation Units go here…]
      </group>
    </body>
  </file>
</xliff>
```

In Business Central translations, the **source-language** attribute is always set to "en-US" because this is the language for text literals in AL code and the default language for Business Central.

The **target-language** attribute must be set to the LCID of the target language, i.e. "de-DE" in the example above.

Finally, the attribute **original** contains the source from where the file has been derived from. In this case, it is the name of the app. However, you could set the **original** attribute of the copied file to "YourApp.g.xlf". But this is not necessarily required since it doesn't fulfill any functional purpose.

2.2.3 Understanding Translation Units

The most important part is within the **group** element of the xliff file: a list of translation units represented by a **trans-unit** element, each of which consisting of the translation of a single text literal. A translation unit looks like illustrated below.

```
        <trans-unit id="Table 36641641 - Field 3461834954 - Property
2879900210" size-unit="char" translate="yes" xml:space="preserve">
          <source>Description</source>
          <target state="translated">Beschreibung</target>
          <note from="Developer" annotates="general" priority="2"></note>
          <note from="Xliff Generator" annotates="general"
priority="3">Table Your Table - Field Description - Property
Caption</note>
        </trans-unit>
```

2.2.3.1 Translation Unit Path

Each text literal is identified by its unique translation unit path which can be found in the **id** attribute of each **trans-unit** element. This path is comprised of all programmatic names in the hierarchy of definition. But instead of using these programmatic names directly, a numeric hash value is generated from each name. The reason for using numeric hash values instead of names is that comparing strings like names is very expensive compared to comparing numbers and hence, loading translations into the user interface would take longer and consume more system resources.

The drawback of using hash values is that readability gets almost lost and it is even not possible to infer a hash to its original name since different names may result in the same hash value. The consequence is, that in case a developer renames e.g. an AL object, all translations of this object seem to get lost because there are no translations available with the new hash value which is based on the new AL object name.

When rebuilding an app after changing the name of the AL object, the translation units referring to the old object name are removed from the .g.xlf file. However, the translations remain in each language's xliff file but they are orphaned which means, that there exists no object anymore referring to these translation units. If you were able to find the hash value of the old name and of the new name, you could use a simple find/replace action to move the orphaned translations to the new object (name).

As this document describes later, Qucamba Reports contains an action that calculates the hash value based on a name that you can enter manually. However, this is would not be the best way to save translations from getting lost. There's also an action for moving existing translations to a new object name. This will be described later in this document.

2.2.3.2 Notes

To help a developer reading the translation unit path, the AL compiler always includes a **note** element with the attribute **from** set to "Xliff Generator", which contains a readable form of the translation unit path where all hash values are replaced by their original name.

Furthermore, there is another **note** element with its **from** attribute set to "Developer". This note is the content of the **Comment** attribute that can be added to all text literals in AL code. For example, a label definition in AL code like

```
var
    DateNotInAllowedRangeErr: Label 'The %1 %2 is not within the
range of allowed dates %3..%4.', Comment =
'%1=Fieldcaption("Posting Date"), %2="Posting Date", %3=min.
date, %4=max. date';
```

would add a note to its translation unit like

```
<note from="Developer" annotates="general"
priority="2">%1=Fieldcaption("Posting Date"), %2="Posting Date", %3=min.
date, %4=max. date</note>
```

A tool helping the developer in the task of translating an app might show this information during editing of translations.

2.2.3.3 Source and Target text

After translating, each translation unit consists of a **source** and a **target** element. The first one contains the original text that the translation unit translates, the latter contains the text translation.

Since the base format of an xliff file is XML, one should always take care of not specifying any characters that are invalid for XML, e.g. an "Ship & Invoice" must be written as "Ship & Invoice".

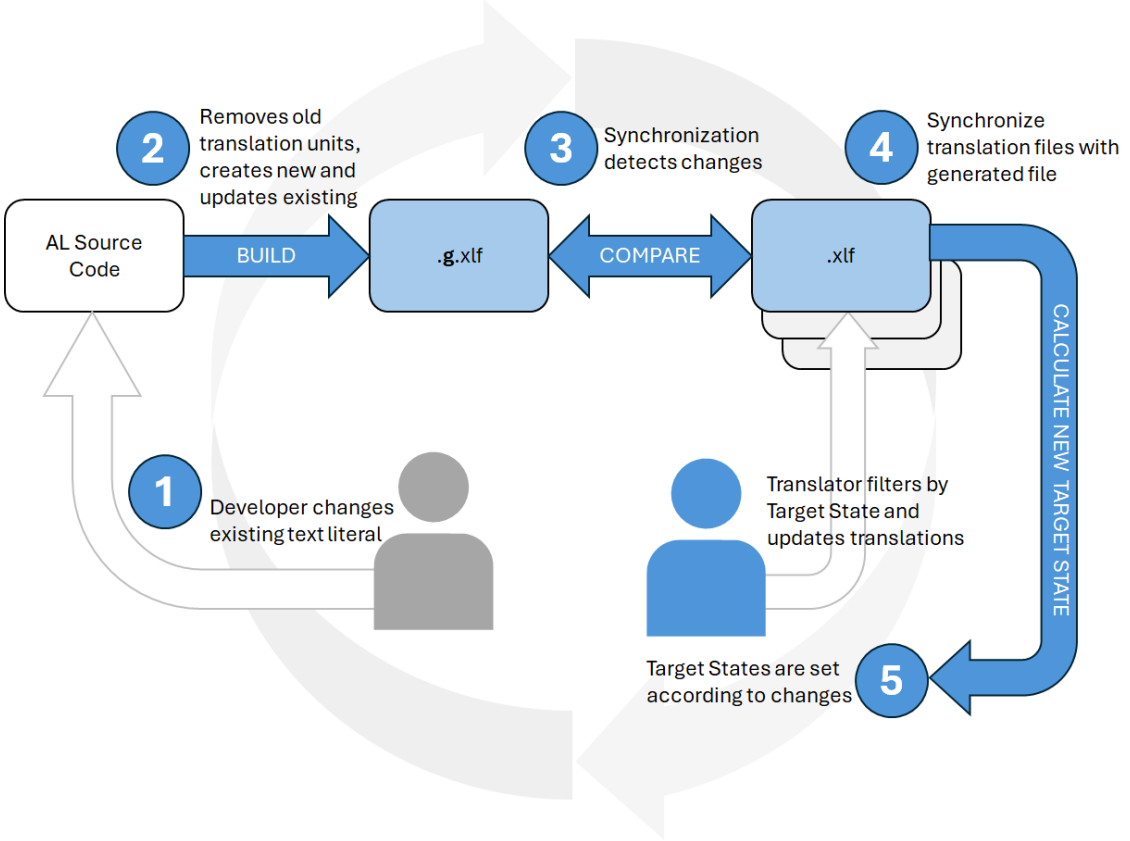
2.2.3.4 Translation State

It is important to understand that when a developer continues to write code after completing the task of translation, existing text literals that have already been translated might change. This could either be an update due to a misspelling or a semantical change with a completely different meaning of the text literal. While the first change can be made hassle-free, the latter change requires a review of all other translations.

This is usually handled through a synchronization mechanism that detects subsequent changes in text literals and escalates these changes into the translation state attribute of all other languages that might require a review. To detect a subsequent change, an additional language file for "en-US" is used. In this case, an app might contain the following translation files:

Filename	Source Language	Target Language	Description
/Translations/YourApp.g.xlf	en-US	en-US	Generated Xliff
/Translations/YourApp.en-US.xlf	en-US	en-US	Xliff for synchronization
/Translations/YourApp.de-DE.xlf	en-US	de-DE	German translation
/Translations/YourApp.fr-FR.xlf	en-US	fr-FR	French translation

When an existing text literal is changed, the subsequent build of the app will transfer the changed text literal to the .**g**.xlf file. However, the .**en-US**.xlf file stays unchanged and thus contains the old value of the changed literal. A synchronization process can now compare all translation units **source** elements from the .**g**.xlf file to the corresponding translation units **target** element from the .**en-US**.xlf file. If these values are not equal, it means that the existing translationhas been changed and thus the state of the corresponding translation units in the .de-DE.xlf file as well as the .fr-FR.xlf file is changed to needs-review-translation as illustrated below.



A translation tool can then draw attention to the required updates in existing translations.

2.3 Conclusion

Translating Business Central apps is very straightforward and Xliff files are a well-designed standard for keeping text literals out of code files. However, until today the common tool chain used for AL development doesn't contain a tool for accomplishing the task of initial as well as ongoing translations. Though Microsoft offers a tool called "Multilingual App Toolkit Editor" (MAT) which lets the user read, edit and write translation files, this tool lacks support for the process of translation beyond simple editing of text.

There have other tools becoming available that are specifically designed for the needs of software development in general as well as for developing Business Central apps in particular. However, these tools have various disadvantages, like they

- are not suitable for the design process in Business Central app development
- work too slow during import and export of Xliff files
- have a user interface (UI) that is too hard to use when writing short (e.g. a caption) and long (e.g. a tooltip) as it is required in Business Central apps

The Qucamba Reports TRANSLATE module is yet another editor for app translations. However, it has some very important advantages over other tools:

- It is specifically designed for the process of Business Central app development
- It has a pretty user interface
- Supports initial translation as well as ongoing changes of existing translations by means of a synchronization action
- Helps in rescuing orphaned translations when renaming AL objects or changing other programmatic names
- Go-to definition action to allow navigating from a translation to its definition in AL source code
- Supports creation of new language files
- Supports automatic translation through Microsoft Azure AI translation services
- Allows the user to manage branch-specific dictionaries
- Helps the developer in creating training material for using trained models in Azure AI translation services to gain a tremendously enhanced translation quality in automatic translations.

Regardless of whether you're going to use Qucamba Reports TRANSLATE or not, it will automatically work in background when working with reports, e.g. to support display of labels in local language. Even when a developer needs to copy an existing report, Qucamba Reports helps by not only copying the report logic and selected layouts but also all translation units of the report. Instead of collecting and editing all parts of the source report by hand, Qucamba Reports offers a Copy-Wizard that lets the developer accomplish this task in just a few seconds.

By the way – despite of the product name, Qucamba Reports TRANSLATE is not restricted to translate reports but lets you translate and edit translations of all other object type in your app as well.

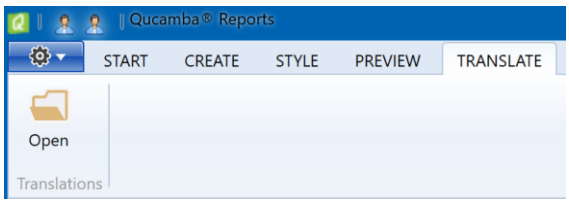
3 Working with the Translation Editor

3.1 Preparing Translation

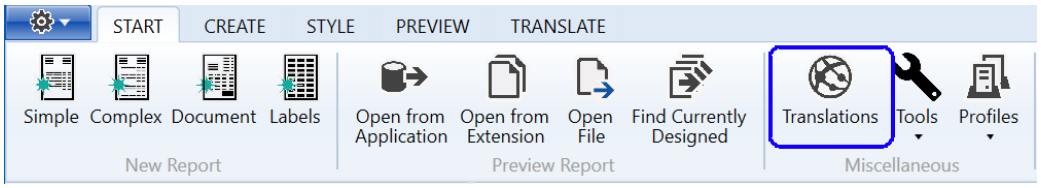
The Translation Manager is available for all Business Central profiles up from BC version 15.0 and resides in its own ribbon menu "TRANSLATE". It can be used to investigate, edit, merge and export translations.

To use the Translation Manager, you need to enable the *TranslationFile* feature in your app's configuration.

Since loading and decoding translations takes some time, the Translation Manager is inactive by default. It can be activated by clicking "Open Translations" in the START ribbon or by selecting the "TRANSLATE" ribbon menu and clicking "Open".



-or-

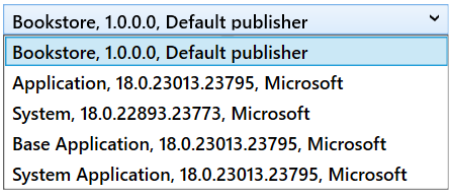


Please note, that you need to activate a profile first before these actions become visible in the application.

3.2 Selecting App and Object Scope

After opening the Translation Manager the master app's translations (i.e. the app the current profile is connected to) are loaded first to allow the user to work on translations while Qucamba Reports continues loading translations of all other dependencies in background. During background translation loading, a progress indicator is displayed besides the app selection drop down list as well as in the application's status bar.

As soon as the loading indicator disappears, other apps can be selected from the drop-down list of apps.



Though all apps of the current dependency graph can be viewed in the Translation Manager only translations of the master app can also be edited. The reason for why all translations are loaded into memory is that these translations are used for additional functionality, like

- Investigating existing objects and their translations
- Copying translations from existing objects to new objects
- Making translation suggests based on existing translations.

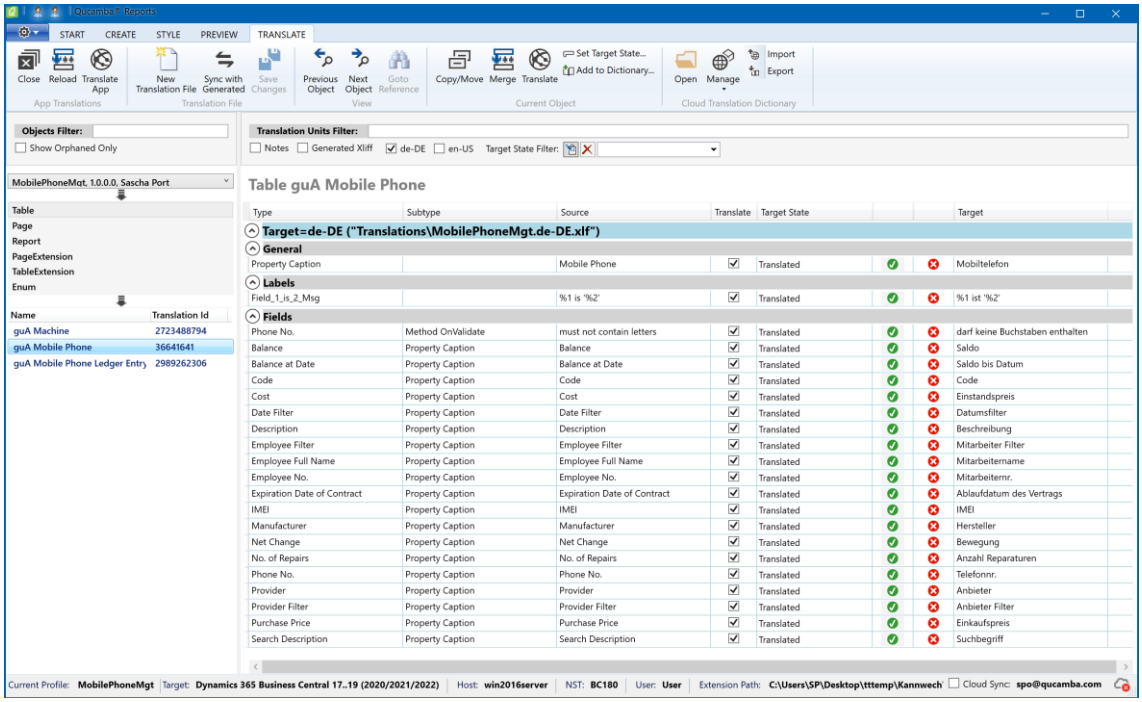
Qucamba Reports usually tries to resolve translation ids to clear symbol names. If for some reason the symbol name is not available, the translation hash value as it is defined in the Xliff files is displayed instead.

During the process of loading translations, Qucamba Reports processes al Xliff files located in each app's "Translations" folder. On dependencies, this folder is retrieved from the app's package file.

Depending on your installation, translations of your base app might reside in a separate app instead of the base app itself. Thus, you should include the translation app as a dependency on your app as well.

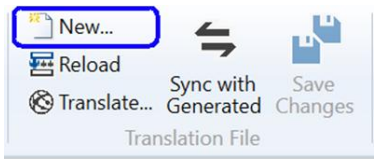
After loading translations, following content is displayed on the left pane:

- a selector for the app to work on,
- a selector for the type of objects contained in the selected app,
- a selector for all objects of the selected app and the selected object type.

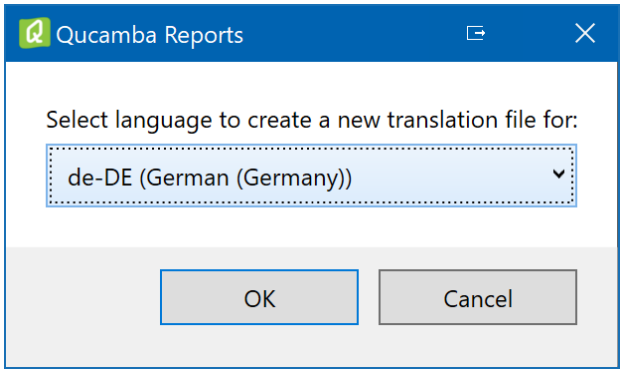


3.3 Creating New Translation Files

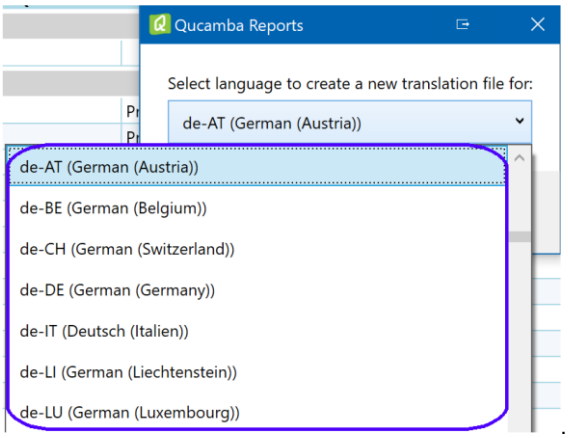
After activating the *TranslationFile* feature in your app's configuration file (app.json), you need to rebuild your app at least once to make the compiler generate the *.g.xlf file in the Translation folder of your app. As soon as the generated Xiff file is updated, load or reload the Translation Manager and choose "New Translation File" from the File menu.



In the Create Translation File window, type the ISO code of a language or select the language from the list and click OK. In this example, we are going to create a new translation file for the language German as it is spoken in Germany.

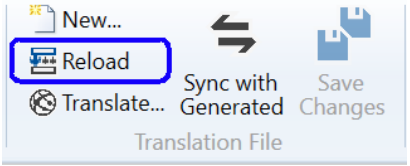


Tip: Just click on the language selection box and start typing the ISO language code e.g. "de-". From here on, you may select from the supported countries that have German as a language



Qucamba Reports creates an appropriate language file and adds it to your app's Translation folder. Usually, you start creating a language file for "en-US". Then, continue creating additional language files, one for each language you would like to support e.g., "de-DE".

Of course, you are alternatively still able to create translation files as you did before by hand. In case you prefer creating Xliff files manually or anytime you edit Xliff files manually, click the "Reload" action located in the File menu of the Qucamba Reports Translation Manager to reflect changes you made to the underlying Xliff files.



For a quick translation you may click the "Translate..." action in the "Translation File" group. This prepares an initial translation of an entire Xliff file after it has been created. However, when editing translations later you should prefer using the "Auto Translate" action of the "Current Object" section instead since this addresses the current object only and gains finer access to translation options.

Note.
Qucamba Reports permanently watches your Xliff files and reloads them as soon as it detects any change. For example, when adding a label in AL code and building your app in VS Code, Qucamba Reports will automatically reload the generated Xliff file and show up the new label. However, to translate the new label you'll need to synchronize your translation files. This process is described in detail in a separate section below.

3.4 The Object View

When selecting an object in the left pane, all attached translations are displayed in the right pane. Each translation (Xliff) file is displayed as a separate group that can be expanded and collapse.

You can now start translating your app or investigating existing translations.

To edit a translation, locate the "Target" textbox for the appropriate translation unit in the target language you want to translate to and directly enter the translation. The "Target" textbox will directly change into edit mode.

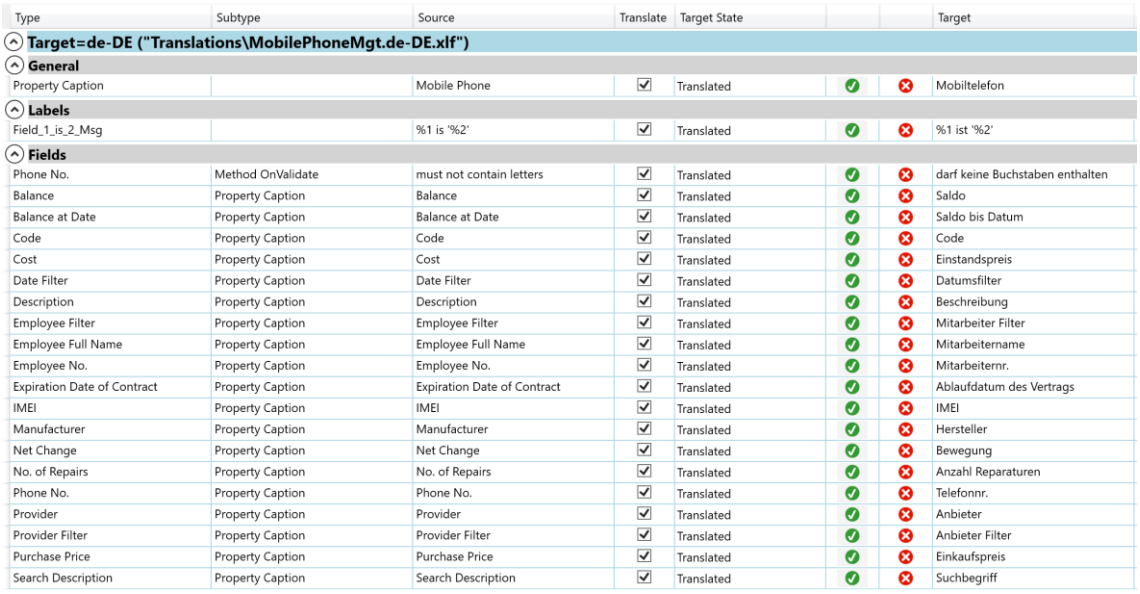
After editing translations, click "Save Changes" to commit your changes to the underlying Xliff files.

Finally, build and publish your app.

Note.
You can always edit the Xliff files directly if needed. However, you should always click "Save Changes" to commit changes in memory to the underlying Xliff files. After editing Xliff files externally, click "Reload" to let Qucamba Reports load these changes into memory.

3.4.1 Columns in the Object View

The object's view allows investigating and editing of symbol translations for the currently selected application object in a data grid as illustrated below.



Type	Subtype	Source	Translate	Target State			Target
Target=de-DE ("Translations\MobilePhoneMgt.de-DE.xlf")							
General							
Property Caption		Mobile Phone	☑	Translated	✓	✗	Mobiltelefon
Labels							
Field_1_is_2_Msg		%1 is '%2'	☑	Translated	✓	✗	%1 ist '%2'
Fields							
Phone No.	Method OnValidate	must not contain letters	☑	Translated	✓	✗	darf keine Buchstaben enthalten
Balance	Property Caption	Balance	☑	Translated	✓	✗	Saldo
Balance at Date	Property Caption	Balance at Date	☑	Translated	✓	✗	Saldo bis Datum
Code	Property Caption	Code	☑	Translated	✓	✗	Code
Cost	Property Caption	Cost	☑	Translated	✓	✗	Einstandspreis
Date Filter	Property Caption	Date Filter	☑	Translated	✓	✗	Datumsfilter
Description	Property Caption	Description	☑	Translated	✓	✗	Beschreibung
Employee Filter	Property Caption	Employee Filter	☑	Translated	✓	✗	Mitarbeiter Filter
Employee Full Name	Property Caption	Employee Full Name	☑	Translated	✓	✗	Mitarbeitername
Employee No.	Property Caption	Employee No.	☑	Translated	✓	✗	Mitarbeiternr.
Expiration Date of Contract	Property Caption	Expiration Date of Contract	☑	Translated	✓	✗	Ablaufdatum des Vertrags
IMEI	Property Caption	IMEI	☑	Translated	✓	✗	IMEI
Manufacturer	Property Caption	Manufacturer	☑	Translated	✓	✗	Hersteller
Net Change	Property Caption	Net Change	☑	Translated	✓	✗	Bewegung
No. of Repairs	Property Caption	No. of Repairs	☑	Translated	✓	✗	Anzahl Reparaturen
Phone No.	Property Caption	Phone No.	☑	Translated	✓	✗	Telefonnr.
Provider	Property Caption	Provider	☑	Translated	✓	✗	Anbieter
Provider Filter	Property Caption	Provider Filter	☑	Translated	✓	✗	Anbieter Filter
Purchase Price	Property Caption	Purchase Price	☑	Translated	✓	✗	Einkaufspreis
Search Description	Property Caption	Search Description	☑	Translated	✓	✗	Suchbegriff

The following table explains the content of each column in detail.

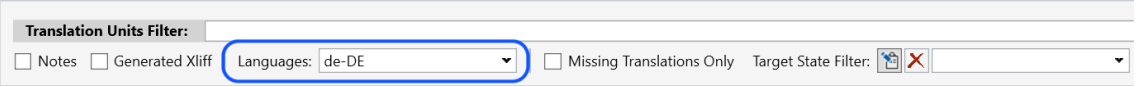
Column Name	Type	Description
Element Name	Editable text	The name of the symbol to translate.
Property	Informational text	Explanation of the symbol kind., usually the name of a property.
Target State	Editable state	The current state of translation as defined by the OASIS specification.
Source	Read-only text	The source text to be translated.
Target	Editable text	The translation of the source text to the target language that is displayed in the group header.
Max. Length	Integer	The MaxLength attribute as applied to the original text literal in AL code or blank if no limit is applied.
Set to Translated	Action	A button that sets the target state to “Translated”.
Set to Review	Action	A button that sets the target state to “Needs Review Translation”.
Notes	Informational text	Developer notes from the Xliff file. Visible only if the “Notes” filter option is checked. Alternatively, notes are also shown in the bottom view of the current translation unit.

3.4.2 Filtering Translation Units

The Object View displays all Xliff files of your app simultaneously. All translation units are grouped by their containing Xliff files and can be expanded or collapsed as needed. By using filters, you can not only reduce the number of translations being shown in the translations view but also investigate which objects and object types contain translations according to your applied filters.

3.4.2.1 *Languages Filter*

You can select which languages to show in the filter header of the Object View. This allows you to turn display of translation files you want to view or edit on and off.

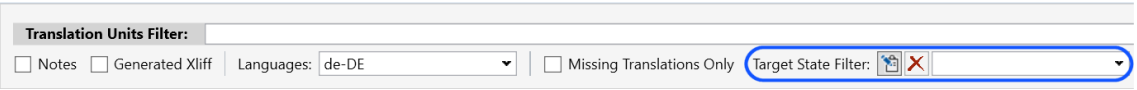


The language code that appears in the filter header bar refers to the target language code of each Xliff file.

> **Tip**
> Sometimes when editing translation texts, you prefer seeing only a single language at a time.
> Instead of selecting und unselecting languages manually, you can also right-click the language code in the list of languages in the drop-down menu to display the language in solo mode.
> When using this selection type, the option for showing the Generated Xliff is cleared automatically.

3.4.2.2 *Target State Filter*

By opening the drop-down of the Target State Filter, you can select the states of translation units being displayed in the translation units view.

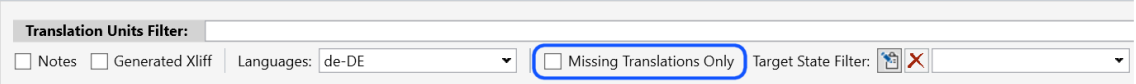


Beside the drop-down list, there're also two buttons. The first button "To do" lets you apply a target state filter that shows all translation units that require any kind of translation, rework or check. The second button lets you clear the entire target state filter.

> **Tip**
> As in the Languages Filter, you may right-click a target state to set the filter to a single target state.Missing Translations Only Filter

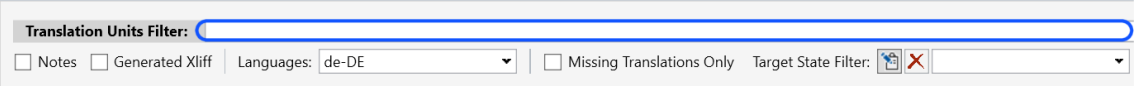
3.4.2.3 *Missing Translations Only Filter*

By checking the "Missing Translations Only" option, the view is limited to show up translation units with empty target text only and regardless of their translation state.



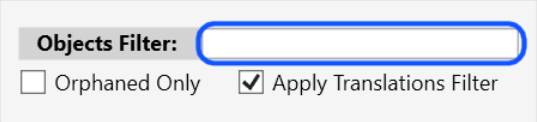
3.4.2.4 Full-text Filter

To apply a full-text filter to the content of translation units, type any text into the Translations Units Filter textbox at the top of the filter bar. This way, you can filter the view to partial content of the source or target column and even to content of the "Notes" column.



3.4.3 Filtering Object Types and Objects

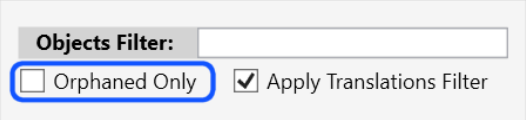
As described earlier, the left panel of the TRANSLATE module shows the current app and its contained objects grouped by object type. By typing text into the Objects Filter, this view can be restricted to objects containing the text entered. The Object Types view will then be updated to show only those object types containing objects within the current filter.



3.4.3.1 Show Orphaned Only

When changing an object's name its translation id hash changes as well. To understand why this happens, read about the basic principles of translation units in the introduction of this document.

In general, orphaned objects and translation units are removed during synchronization. However, you might want to manually migrate orphaned translation to the new object's name. In this case, activae the Orphaned Only filter to let the Object view show objects with orphaned translations only.

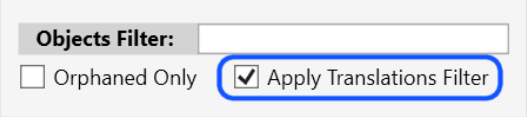


Then, walk through each object in the filter. Then for each object, select all translation units and use the **Copy/Move Translations** action to assign the new object name.

Finally, save your translations and synchronize.

3.4.3.2 Applying Filters to the Object Types and Objects views

Another powerful feature is the ability to restrict both the object type view and the object view to entries that are included in the currently active Translation Units filter. This provides a clear and comprehensive overview of all objects and their corresponding translations.

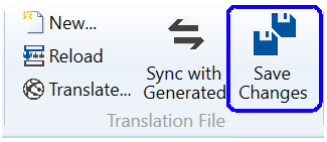


When this option is enabled, the left panel is automatically updated whenever you change a filter value.

The Apply Translations Filter is active by default. However, depending on your filters it may happen that the current object you're working on goes out-of-scope resulting in losing focus on the current object. To avoid this during editing, simply deactivate the "Apply Translations Filter" switch.

3.4.4 Saving Changed Translations to the Underlying Xliff Files

When editing translations in the Translation Manager, changes are not committed to Xliff files unless you click the "Save Changes" action.



The "New Translation File" action allows to add a new language to your app. Though this could also be accomplished from within the app itself but adding a new language in Qucamba Reports is an even easier task. However, each time translation files are added, changed or removed directly in the app, Qucamba Reports needs to reload translation files to reflect these changes. Editing of translation files is usually recognized by Qucamba Reports an the changes are loaded automatically. However, changes that are not reflected automatically need to be reloaded by clicking the "Reload" action from the Translations ribbon group.

3.5 Advanced Editing in Object View

The Qucamba Reports TRANSLATE module is designed to offer the designer a great user experience when translating an app. The main advantage above other translation tools is, that Qucamba Reports is specifically designed to meet the specific requirements of a Business Central developer. Various feature help, completing the task of translating an app as easy as possible.

3.5.1 Translation State

The translation state is part of the OASIS standardized Xliff format. Following table gives a brief description of the states defined in the Xliff format specification[3]:

final	Indicates the terminating state.
needs-adaptation	Indicates only non-textual information needs adaptation.
needs-l10n	Indicates both text and non-textual information needs adaptation.

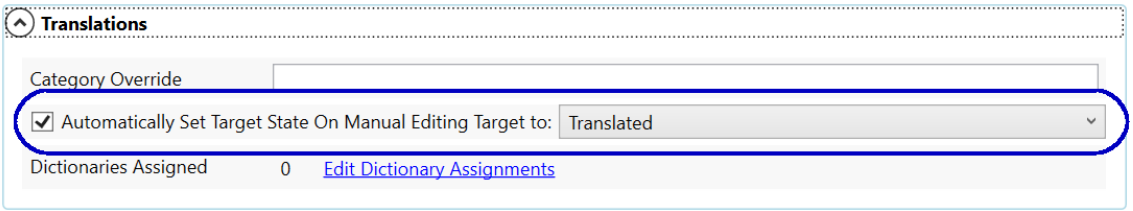
[3] XLIFF Version 1.2 OASIS Standard (https://docs.oasis-open.org/xliff/v1.2/os/xliff-core.html#state)

needs-review-adaptation	Indicates only non-textual information needs review.
needs-review-l10n	Indicates both text and non-textual information needs review.
needs-review-translation	Indicates that only the text of the item needs to be reviewed.
needs-translation	Indicates that the item needs to be translated.
new	Indicates that the item is new. For example, translation units that were not in a previous version of the document.
signed-off	Indicates that changes are reviewed and approved.
translated	Indicates that the item has been translated.

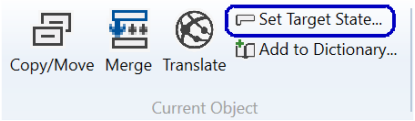
In Qucamba Reports, an additional value "unknown" is used in cases when a translation state is not provided by a translation file.

Usually, when translating Business Central apps, only the target-states **needs-translation**, **needs-review-translation** and **translated** are being used. However, you can use the other states as well.

When editing a translation unit's target text, Qucamba Reports can automatically set the translation state to a new state, which in most cases is the state "translated". Thus, while walking through translation units and entering the target text manually, the state of the translation unit will change to the new state. This behaviour can be activated and customized in the profiles settings under the Translations section as illustrated below.

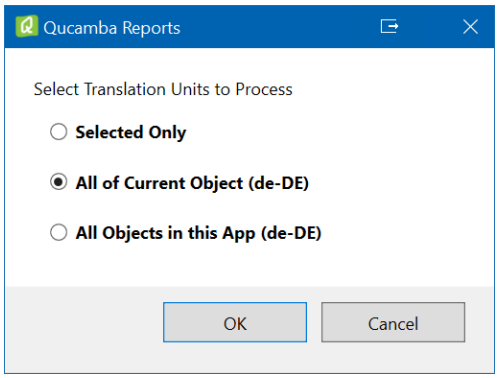


When a new language file is created or when moving or combining translation files, it might become necessary to change the translation state of a bunch of translations. You can do so by using the "Set Target State" action from the ribbon bar.

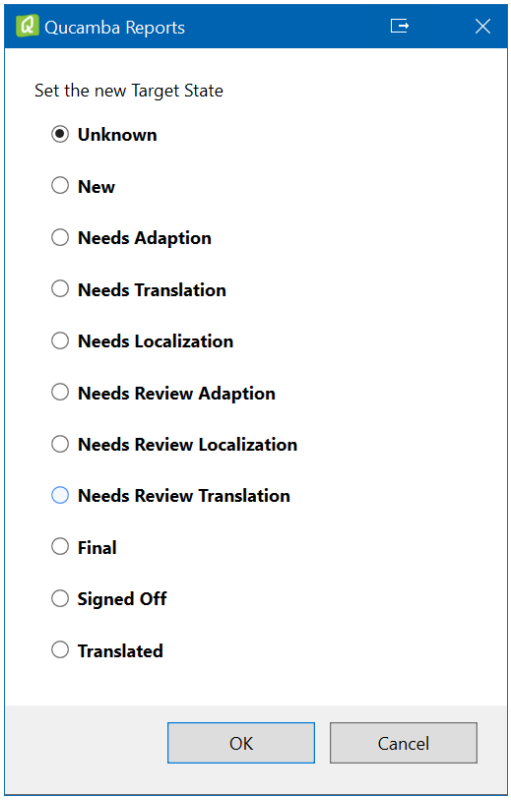


This action allows you to either set

- the state of the currently selected translation units only (i.e. those lines you select by holding down the Ctrl and/or Shift keys when clicking),
- or all translations *for the current object* of all currently visible languages as selected in the header bar
- or even all translations *for all objects contained in the app* of all currently visible languages as selected in the header bar.



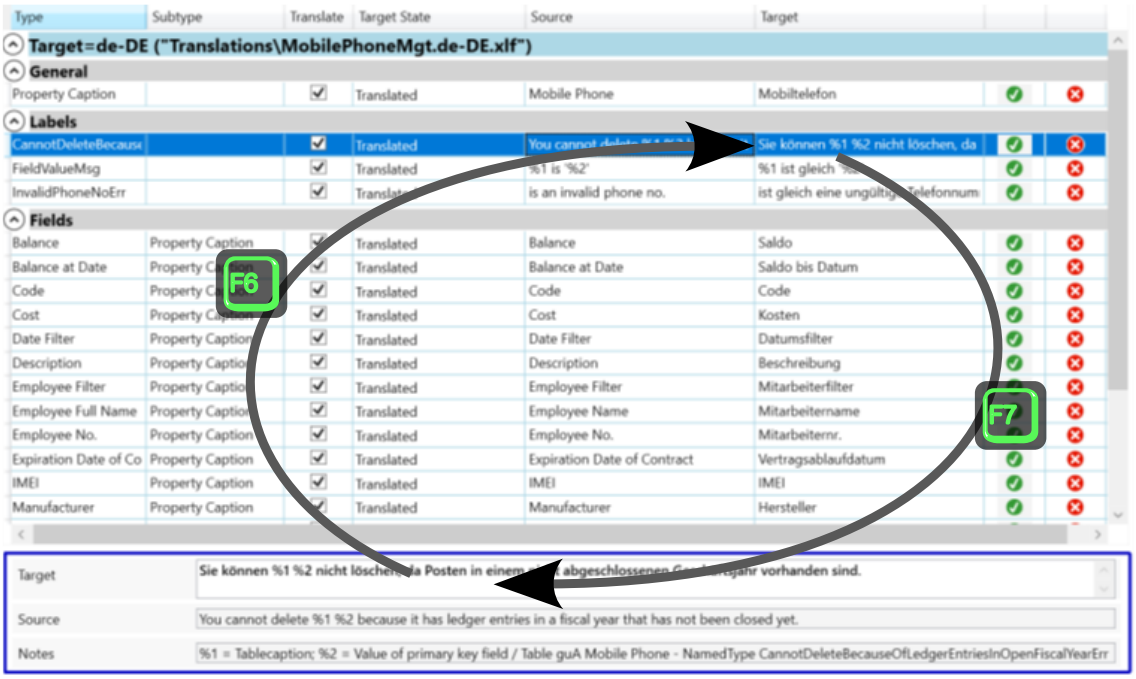
After selecting the scope of operation, Qucamba Reports asks for the new state to set as the target state.



Please note, that changes are not written directly to the underlying Xliff files. To commit changes to the files, you always need to click "Save Changes" in the ribbon menu.

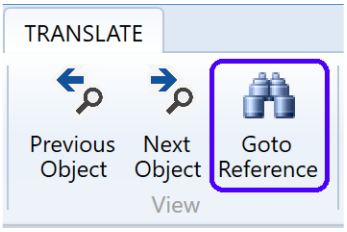
3.5.2 Bottom Translation Unit View

Most of the translations a Business Central developer needs to type are captions such as field captions, object captions, actions captions. These captions tend to contain short texts. However, properties or labels have longer text, e.g. tooltips. For editing long texts, the target text box in the data grid is uncomfortable. Instead, these texts can be edited in the bottom translation view which becomes automatically visible when a single translation unit is selected. Additionally, the Source text as well as developer notes are being displayed. Instead of using the mouse to set the cursor in the bottom target textbox, simply press the F7 key. Vice versa, use the F6 key to jump back to the target textbox within the data grid.



3.5.3 Go-to Reference

When a translation unit of the current master app (i.e. the app you're working on by means of your current Qucamba Reports profile) is selected, the Go-to-Reference function in the "View" ribbon group allows you to investigate the symbol that the translation unit refers to in Visual Studio Code.



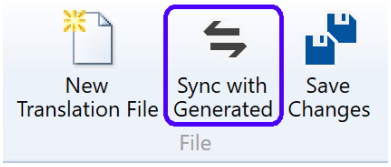
By calling the Go-to-Reference function, your app workspace is opened in Visual Studio Code, the file that contains the related AL object is opened and the cursor is placed near the definition of the symbol.

Tip: The shortcut for this action is the same shortcut as it is in Visual Studio Code (and others): <F12>.

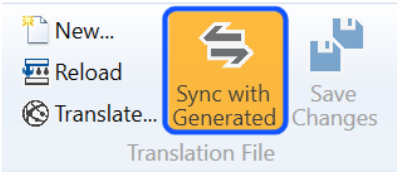
3.6 Synchronizing Translation Files With Generated Xliff

When working on your app, you might add additional string literals that will be included in the *.g.xlf file by the compiler automatically. However, the AL Language compiler does not update the language files accordingly.

Qucamba Reports allows you to synchronize the generated Xliff file with each translation file by selecting "Synchronize with Generated Xliff" from the Tools menu.



Additionally, when Qucamba Reports recognizes any change in the content of the generated Xliff file it lets the "Sync with Generated" action button blink to inform you that there are pending changes to be synchronized.

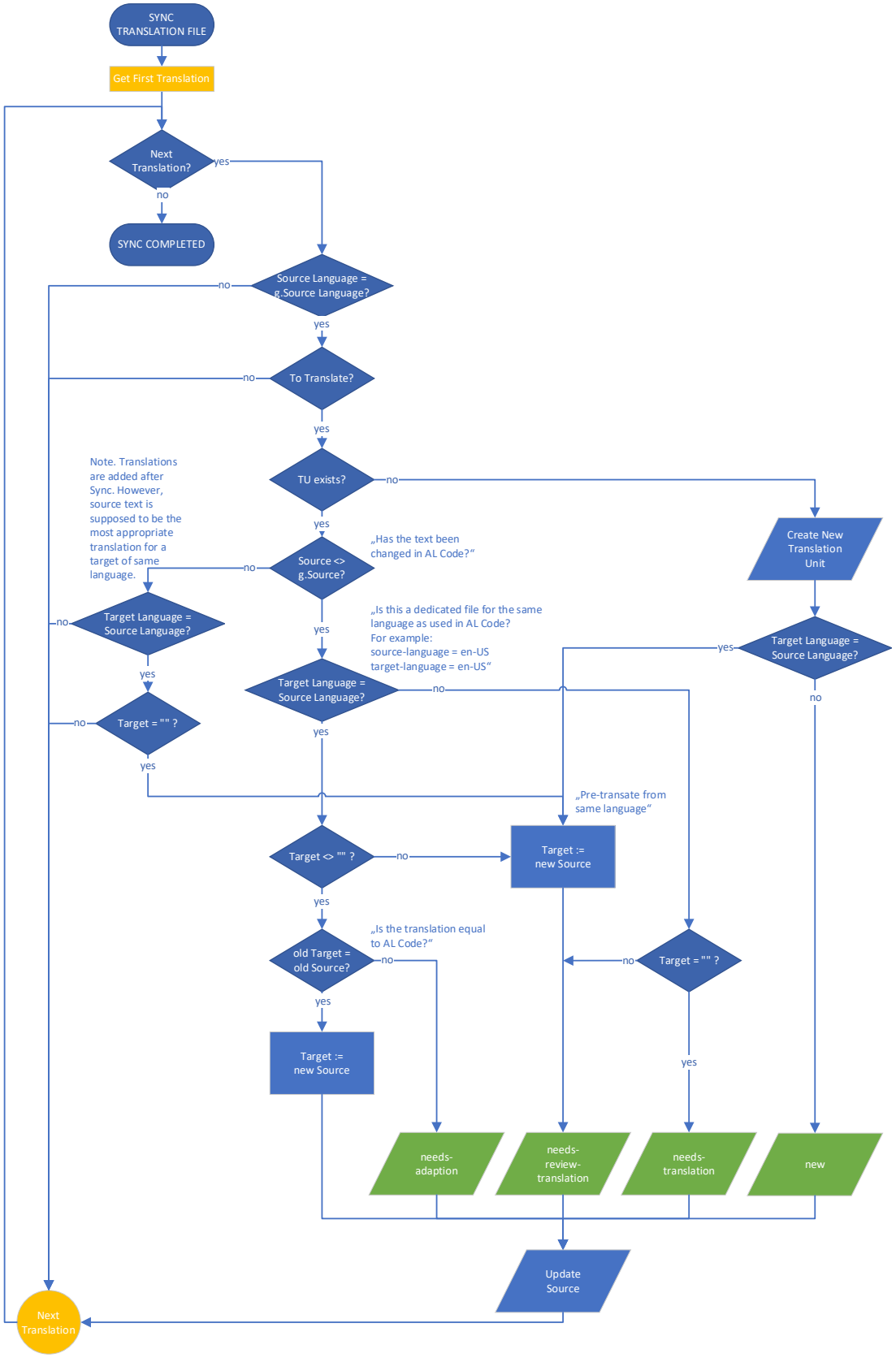


Usually, you would then go and complete your current translation tasks, save your translation files and then click the "Sync with Generated" action button to synchronize all outstanding changes. The button stops blinking as soon as synchronization is completed and no more pending changes need to be synchronized.

By performing a synchronization of all language-specific translation files with the generated Xliff

- All orphaned translations are permanently deleted from all translation files.
- Missing translation units are added to each translaton file.
- Existing translations are synchronized.
- The target state is set appropriately.
- The translation files will be recreated in the same order as the generated Xliff file.

The following illustration helps understanding how the synchronization process works and how the target state changes during synchronization.



Despite of setting the target translation states as described in the flow diagram above, there are some cases in which the system decides to set the target state differently. For example, when captions contain the ampersand character ("&") it is not clear whether it is meant to be a semantic "et" or a designator to let the subsequent character act as a (yet legacy) shortcut key. Thus, the system sets the target state to "needs-adaptation" when translated automatically. However, when entered manually, the target state will revise to "translated".

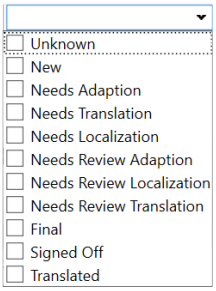
To avoid loss of existing translations of unintentionally orphaned translation units, we recommend creating backups of your translation file before performing any of the Translation Manager actions as well as using source control for Xliff files as well. The process of handling orphaned translations is described in detail in the next section.

When the sync operation is completed, all changed translation files are saved back to their origin on disk. After this, the person who translates can continue with the process of translating the app.

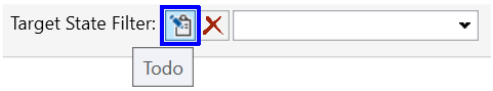
At this point, the translations data grid uses colors to visualize the translation unit's target state as follows.

Blue	**New translations, not yet translated.** A translation unit has just been added and requires first-time translation.
Red	**Translations that need adaptation.** An existing translation has been changed and might have even changed semantically. Some of them may just need to be inspected and confirmed. Others might need new translation.
Yellow	**Adaptation or Localization needs to be reviewed** A translation might be fully translated. However, it could possibly require some adaptation or localization.
Green	**Translations need review** A translation has almost completed. It might have changed slightly but is most likely correct and just needs to be confirmed.

You can set the view to be filtered for one or multiple target states by using the filter combo box in the header bar as illustrated below. Place a checkmark in all filters that should be contained in the view.



Note. *You can also set a filter that automatically includes all translation units that require your attention by using the "Todo" button in the filter bar.*



The Todo filter can further be customized by checking or unchecking each state in the combo box.

3.7 Handling Orphaned Translation Units

The previous section described the process of synchronizing translation units in detail. However, there may be translation units left in the end which became orphaned due to changing an object's name or an element's name. This happens, because whenever a name is changed its hash value changes as well. The process of synchronizing translation units synchronizes translation units that share the same translation unit path (or "id") which includes comparing the name-based hash values.

To understand the subsequent process after synchronization, it is essential to have a basic understanding of the translation unit path. The translation unit path is represented by the "Id" attribute of a single translation unit in the Xliff file and consists of multiple steps, a sequence of pairs, each containing a translation symbol kind and a hash value. The translation symbol kind defines the type of the respective hash value that follows. For example, in the translation unit path step "Table 47114711288," "Table" is the translation symbol kind, and the value 47114711288 represents the hash value. Since the hash value is computed from a text literal such as an object's name or an element's name, two translation unit paths are not the same after changing any of the referred names. The result is a translation unit that is contained in the translated Xliff files but not in the generated Xliff file. Therefore, these translation units are called orphaned.

To avoid loss of orphaned translations when an object's name or element's name has changed, Qucamba Reports (starting with version 8.0) uses an after-sync post processing that takes all translation units that have been newly created during the sync and tries to find an orphaned translation unit with the same translation unit path and the same source text. However, rather than comparing the entire translation unit paths, only the symbol kinds are being compared while leaving out the hash values. If both match each other, the target text is transferred to the new translation unit and the target state is set accordingly. If the orphaned translation unit has the target state "translated", the new target state will be set to "NeedsReviewTranslation". On the other hand, if the

orphaned translation unit's state is different from "translated", the new target state will be set to the same value.

The result is that most changes in object or element names can be seamlessly resolved without any loss of translations and without any user interaction. However, all other remaining translation units will be discarded after synchronization.

3.8 Translating Objects and Navigating Through Translations

When the app that you are currently working on by means of the Qucamba Reports profile is selected in the Translation Manager window, translations can be edited.

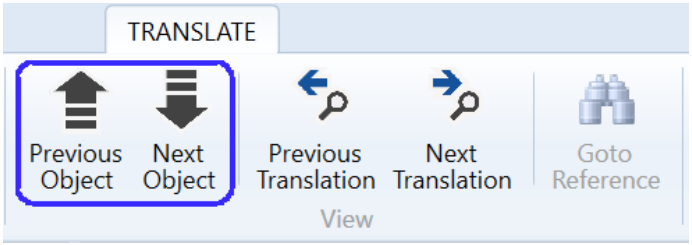
To edit a translation, place the cursor in the Target column and enter the translation.

Type	Subtype	Source	Translate	Target
Target=de-DE ("Translations\Bookstore.de-DE.xlf")				
General				
Property Caption		Book	☐	Buch
Fields				
Author	Property Caption	Author	☐	Autor
Author Provison %	Property Caption	Author Provision %	☐	Autor Provision %
Blocked	Property Caption	Blocked	☐	Gesperrt
Created	Property Caption	Created	☐	Erstellt
Date of Publishing	Property Caption	Date of Publishing	☐	Veröffentlicht
Description	Property Caption	Description	☐	Beschreibung
Edition No.	Property Caption	Edition No.	☐	Auflagenr.

If the currently focused cell is not yet in edit mode, you can either press <F2> to enter its edit mode or use the mouse to left click on the cell. Use the <Enter> key to accept the entered translation and to proceed with the next translation, i.e. the next line. Again, use the <F2> key, to enter the cell's edit mode.

Tip: If you want to overwrite the entire content of the focused cell you do not even need to press <F2>. Instead, simply start typing the new content.

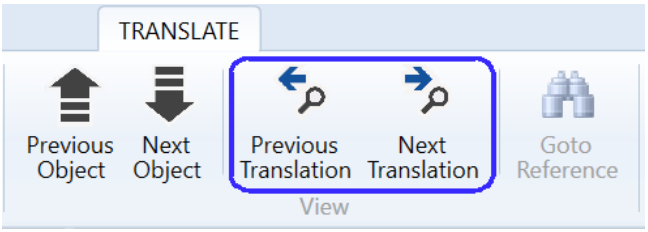
After reaching the end of the current object, you may want to proceed with translating the next object of the current or even the next object type. You can navigate through all objects regardless of the object type by pressing the keys <Ctrl> + <PageUp> to go to the previous object or <Ctrl> + <PageDown> to go the next object. You can use the mouse as well and click the appropriate action in the "View" ribbon group.



However, after translating the app for the first time you don't want to walk through all translation units that have already been translated. Instead, there may be just a few new translations and even

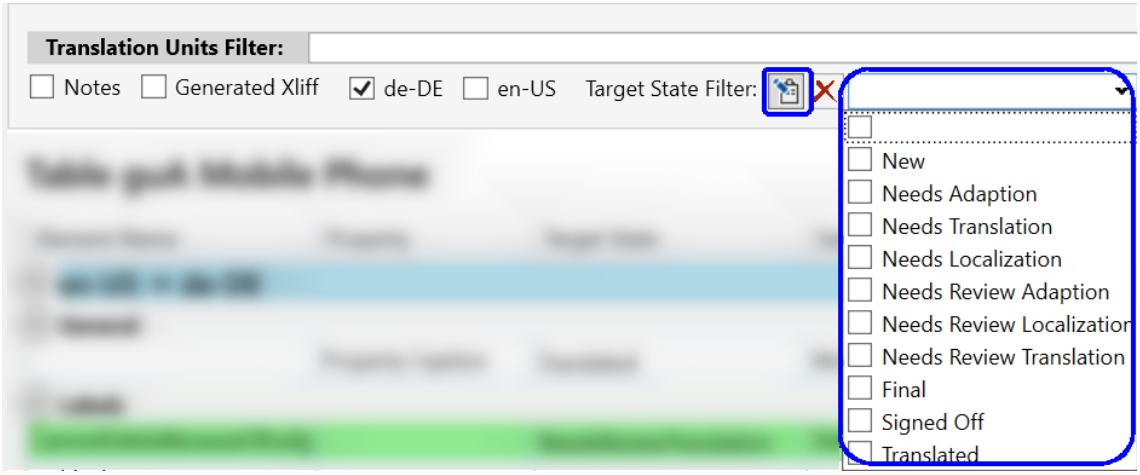
changes in existing translations. Usually, the process of synchronization is used to update each translation's state. Which is described later in this guide.

To find only those translations that require your extension, you can use the Previous/Next Translations action buttons instead or use the according keyboard shortcuts <Ctrl> + <Up> to go to the previous translation or <Ctrl> + <Down> to go to the previous translation. Note, that navigation only stops on translation-units that require your attention.



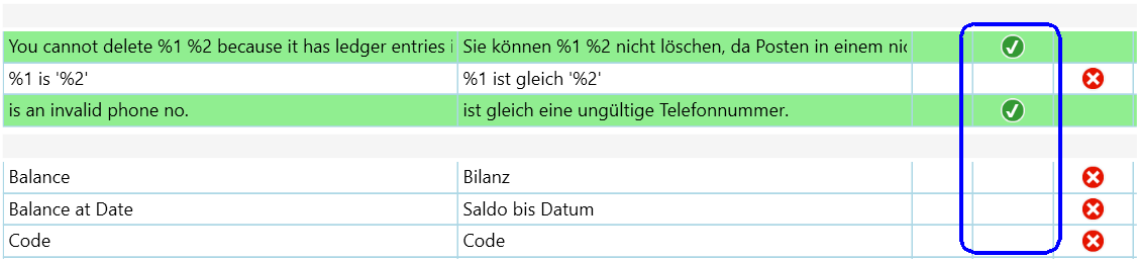
Furthermore, you can set a filter in the filter header of the translation units view.

You can either filter by particular states by placing a checkmark in the Target State Filter combobox on each state. Or you can use the "Todo" filter button to activate a filter to all translation states that need to be addressed.



Also, you can type in a Translation Units Filter in the appropriate text box above to apply a full-text filter to the translation units and to navigate only through translations within this filter. However, navigating while a Translation Units Filter is applied might be slow.

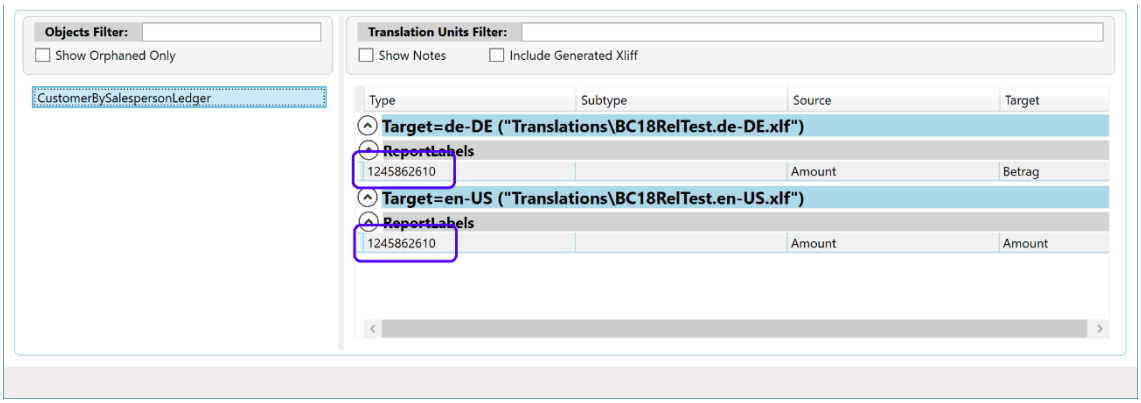
When using a translation's shortcut button to set its state to "translated", the translation editor moves the focus to the next translation automatically.



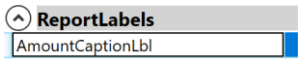
Note, that your changes are not committed to the underlying Xliff files unless you click the Save All Changes action.

3.9 Changing an Object's Name

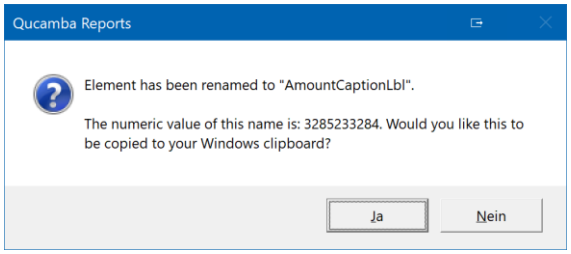
When loading an app's translations, the Translation Manager tries to resolve all numeric translation path values to symbol names allowing you to understand what you are translating. However, you might recognize that some objects and/or symbol names appear orphaned i.e., the Translation Manager displays a numeric value instead of the object's or symbol's name. This usually happens after renaming a symbol that had already been translated.



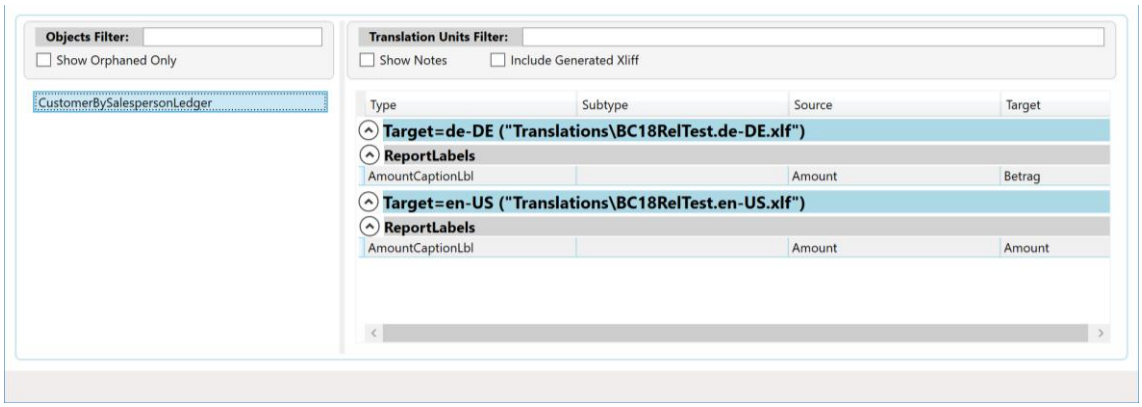
The Translation Manager helps you in getting orphaned entries re-assigned. For example. to re-assign translation units to a renamed field, simply type in the correct field name in the Type column.



To reassign these translation to an object, simply enter the object's name (without the object type) into the appropriate column. Based on the name you entered, Translation Manager calculates the numeric translation value for the object and updates all translation units of the selected object in all translation files accordingly.



This results in updated translation units as illustrated below.



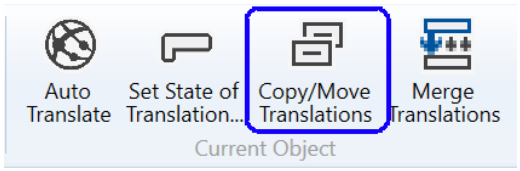
After clicking the Save All Changes button, changes are written back to the underlying Xliff files.

Note: To re-assign translation units of an entire object, use the "Move Translations" function described below instead.

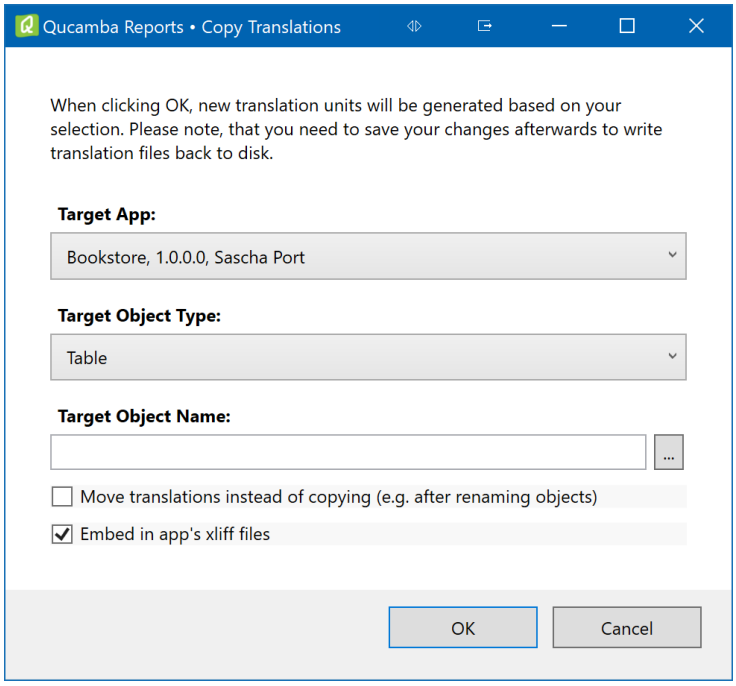
3.10 Copy or Move Translations to Objects

In some cases, you might want to *re-assign all* translations of an object to another object or you might want to *copy all* existing translations of an object to a new object. Typically, this happens after copying a base object, e.g. a report from the base app that you would like to substitute.

To accomplish this, click on "Copy/Move" located in the "Tools" ribbon group.



This opens the "Copy Translations" window as illustrated below.



As you can see in the illustration above, the Target App is already preselected with the current master app, i.e. the current app by means of the currently active profile.

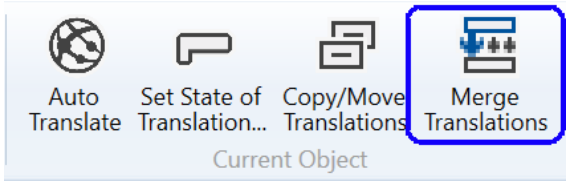
Next, you can select the new target object type. This means, that you can even move or copy translations between deferring object types.

Finally, you select the target object from the list or simply enter the target object's name.

Usually, you should leave the "Embed in app's xliff files" checkbox checked to write the changes to the underlying Xliff files. If you uncheck this option, separate Xliff files containing the target object's name will be created. However, since you can only have one Xliff file per target language in your app, these separated files would have to be merged to the main Xliff files.

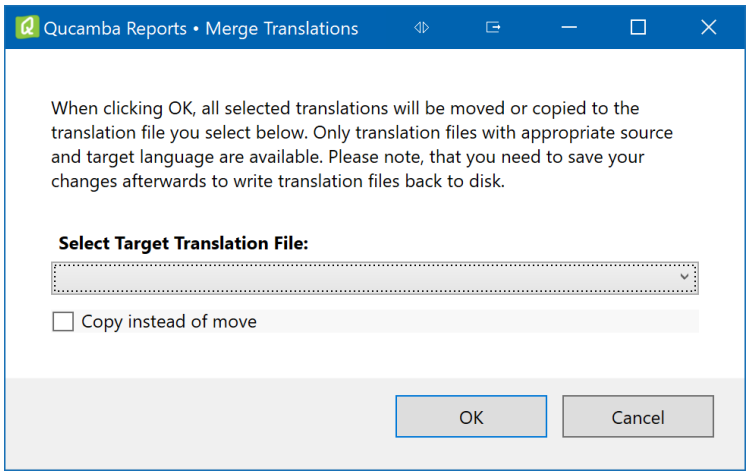
In case you need to merge translations for some particular objects from separate files into the app's Xliff files, there's also a Merge operation available in the Translation Manager.

First, select the translation units you want to merge to the app's translation files. Then, click "Merge" in the "Tools" ribbon group. If you do not have multiple translation units selected, Qucamba Reports asks you, if it should merge all translation units of the current object or just the selected one.



This opens the "Merge Translations" window that let's you pick the target translation file. Usually, this is the one Xliff file of the app that provides the same source language as well as the same target

language and that is not the generated Xliff file. Thus, most often there's just a single choice contained in the "Select Target Translation File" selection box.

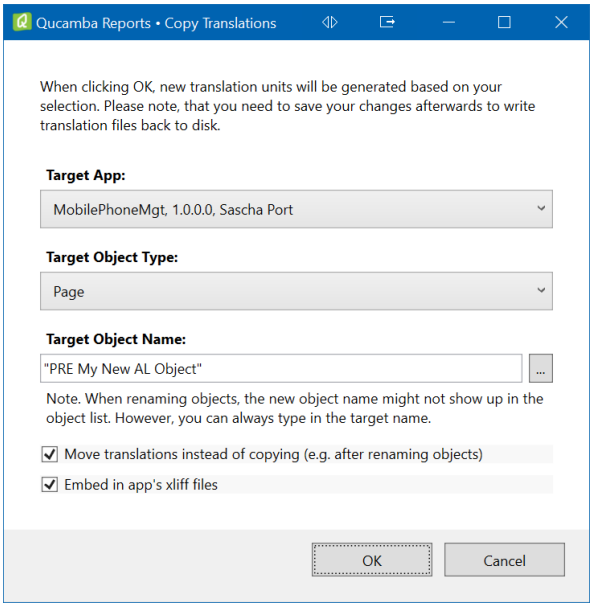


After clicking OK, Qucamba Reports merges the selected translation units into the selected Xliff file. However, remember that you usually also want to remove the merged files from your app's workspace.

3.11 Renaming AL Objects Without Translations Getting Lost

1. Rename the object in AL
2. Rebuild the app in VS Code (Ctrl + Shift + B)
3. Examine that the new object name appears in Qucamba Reports translation editor. The new object contains all generated translations (.g.xlf) while the old object name still persists with the old translations. Be aware that after reloading translations, you might not see the old object name anymore. Instead the object is shown as an "orphaned" object. Use the "Show Orphaned Only" checkbox to filter the objects view to orphaned objects. This way, it's easier to find the renamed object.
4. Switch all languages in to view by activating each language's checkbox in the translation units header bar.
5. In Qucamba Reports, click the Copy/Move action in the ribbon menu.
6. Select the new target object. It the new name is not shown in the object picker list, type in the correct target object name manually. Also, don't forget including your object name prefix. As the prefix is part of the object name, it will affect the object name's translation hash value as well.
7. Make sure, the "Move translations instead of copying" option is checked.
8. Make sure, the "Embed in app's xliff files" option is checked.
9. Finally, click to move the translations of the old object into the new.
10. After copying the orphaned translations, it's strongly recommended to reload translations by clicking the "Reload" action in the ribbon menu. Reloading translations will save all changes

to the underlying xliff files and reload translations so that the orphaned -and now empty- objects disappear.

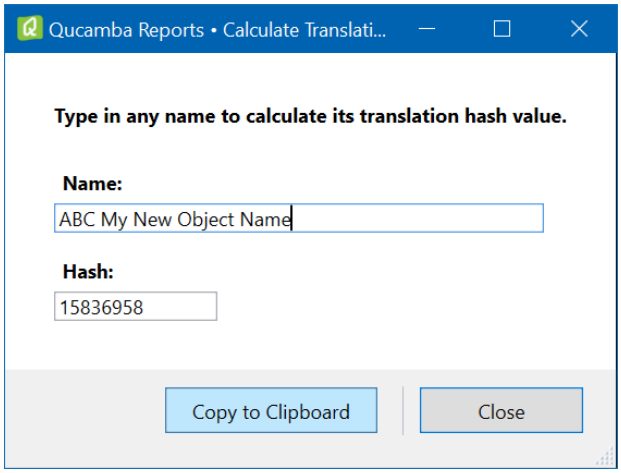


3.12 Translation Hash Calculator

In some situations, you want to edit the Xliff files directly and change some names manually without losing related translations. For example, when changing names of labels after they had already been translated, you might want to go directly into your translation files to replace the old translation id hash value with the one based on the new label name.

For these purposes, Qucamba Reports lets you calculate the translation id hash value based on a name that you type in.

To open the Translation Id Hash Calculator, click “Hash Calculator” in the TRANSLATE ribbon menu or press F11. Also, the calculator can be found in the “Tools” button on the START ribbon menu. Thus, it can even be used without activating the TRANSLATE module.



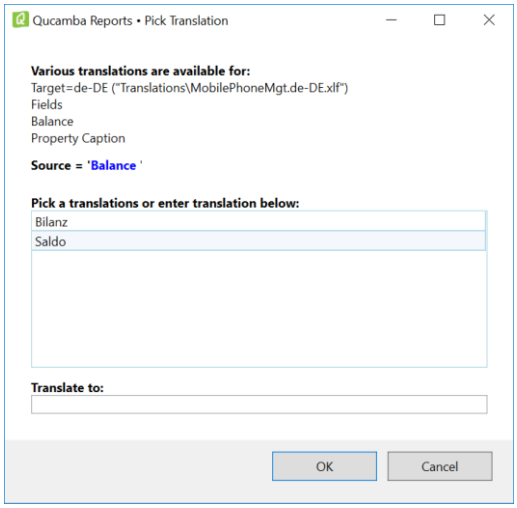
After typing in the entire name to calculate the hash for, click on "Copy to Clipboard" to have the hash value copied to the Windows clipboard.

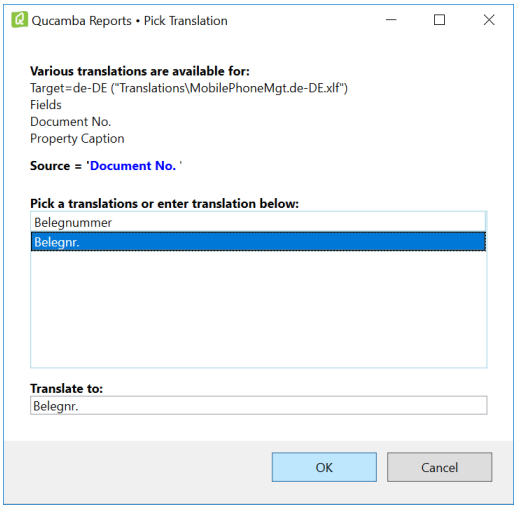
> *Note. Instead of changing a translation id hash manually, you could also edit the old translation name. This automatically calculates the new hash id.*
>
> *Furthermore, when changing an object's name you can transfer all its translations to the new object (name) by using the "Copy/Move Translations" action located in the "Current Object" section of the TRANSLATE ribbon menu,*
>
> *However, in both cases you need to apply the transfer of translations before performing synchronization, because this would remove the old and orphaned name with all of its translations.*

3.13 Auto Translation

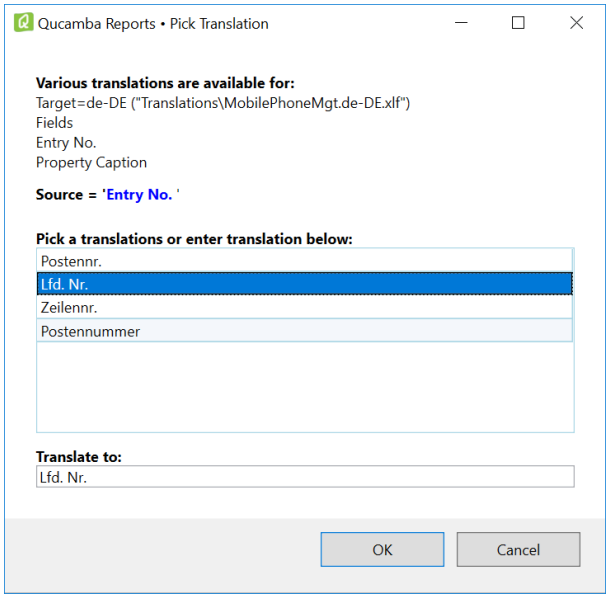
When translating an object, you should usually start by clicking the "Auto Translate" action. Qucamba Reports then looks up all yet untranslated texts by searching for the same source text within all apps contained in the dependency graph.

In case different translations are found, Qucamba Reports opens the "Pick Translation" window and allows the user to either pick the most appropriate translation and/or enter the desired application manually. The following illustration shows how a field called "Balance" is translated automatically.





In case there is no translation of the same source text contained in any of the apps within the dependency graph, Qucamba Reports starts to search for all existing elements with the same translation path but by ignoring each object's translation id. This means, that e.g. for translating the "ToolTip" property of a page field called "Blocked" all pages are searched for a field called "Blocked" containing a translated property called "ToolTip". Again, Qucamba Reports opens the "Pick Translation" dialog.



In some cases, the list of possible translations might already contain a perfect translation. However, you can pick the most appropriate translation and edit it int the "Translate to" field before accepting the translation by clicking OK.

If you configured the Text Translation Service as described alter in this guide, all translations that couldn't be resolved from existing translations will be sent to the translation service.

All remaining texts that are still empty need to be translated manually by clicking into the cell within the "Target" column and typing in the translation.

Usually, no untranslated units will remain after using the translation service. Depending on the quality of translations you'll need to review some or all of these translation units. The translation state of each translation unit will be set according to the source of translation and the supposed quality reported by the translation service.

After all texts of the current object have been translated, use the "Next Translation" action in the View menu to navigate to the next object of the app.

While working with the Translation Manager you might also find some keyboard shortcuts to be helpful. You can find the assigned keyboard shortcut in the tooltip of each action item. Also, the Keyboard Shortcuts chapter in the end of this document lists the most important shortcuts of the Qucamba Reports TRANSLATE module.

4 WORKING WITH DICTIONARIES

Dictionaries are used to improve the quality of automated translations. Qucamba TRANSLATE uses dictionaries both during automatic translation and to generate training material for Azure Translator. This significantly improves the quality of translations that do not have an exact match in the dictionaries.

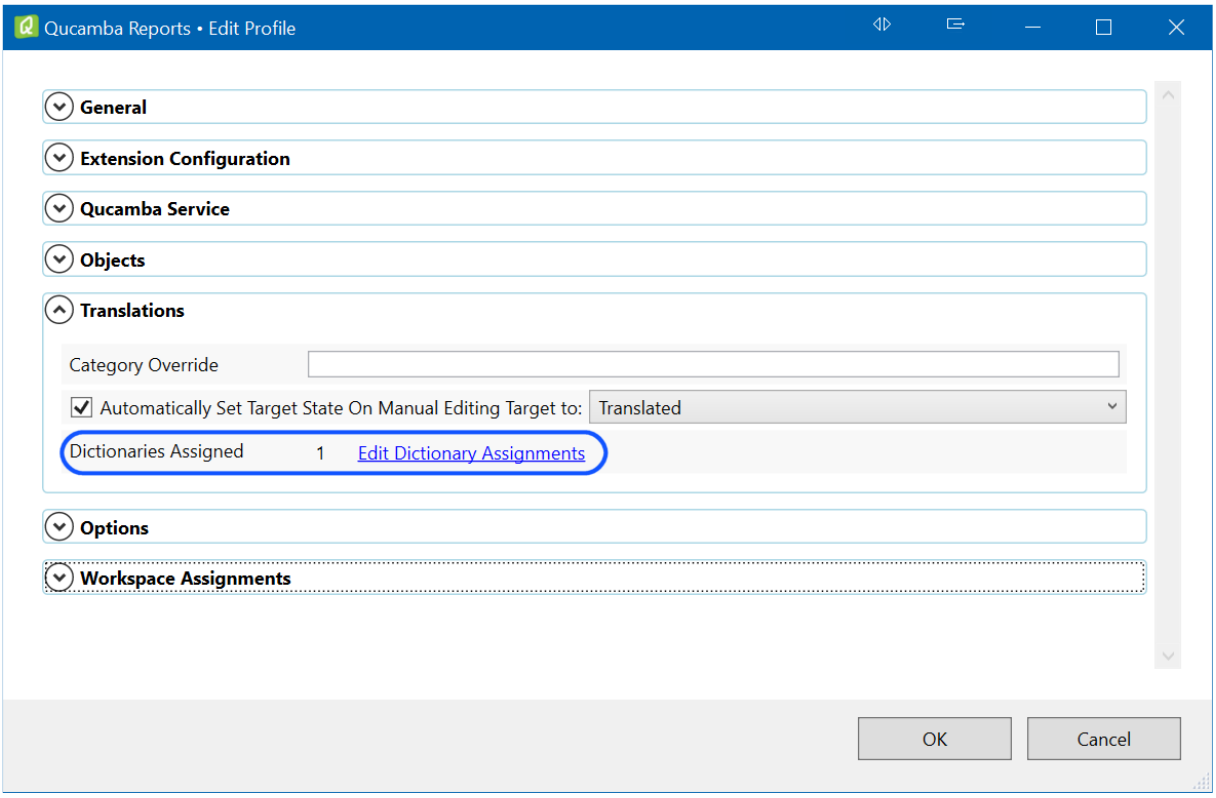
The primary purpose of a dictionary is therefore to translate industry-standard or technical terms more accurately and, above all, consistently.

The role of dictionaries in automated translation is described later in this document. This section first focuses on the general principles of how dictionaries work.

4.1 GENERAL INFORMATION ABOUT DICTIONARIES

All dictionaries are stored and managed in the cloud. This means that all users with the same Qucamba company license can access and collaboratively edit the same dictionaries and their content. An online connection is therefore required when working with dictionaries.

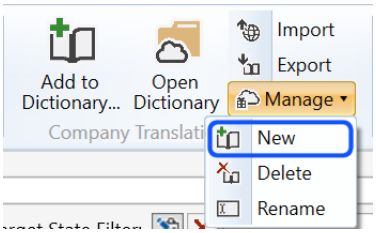
For a dictionary to be used during the translation of an app, it must be selected in the profile associated with that app. To do this, open the desired profile for editing and click **Edit Dictionary Assignments** in the **Translations** section.



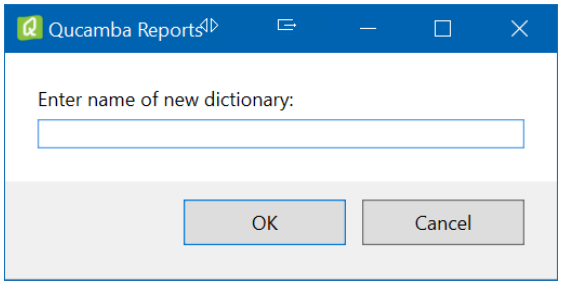
When assigning dictionaries, you can reorder the entries in the list via drag and drop. This defines the priority of dictionary use during dictionary-based translation. The topmost dictionary in the list is selected by default whenever a dictionary selection dialog appears, such as when adding entries from the translation view.

Creating a New Dictionary

To create a new dictionary, go to the **Company Translation Dictionaries** ribbon group, click **Manage**, and select **New**.

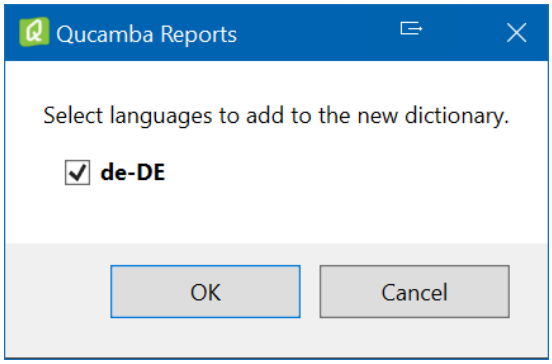


In the dialog that appears, enter the name of the new dictionary.



The name of your app is suggested by default, as dictionaries are typically used to translate domain-specific terms from your app. However, you may also create cross-industry dictionaries.

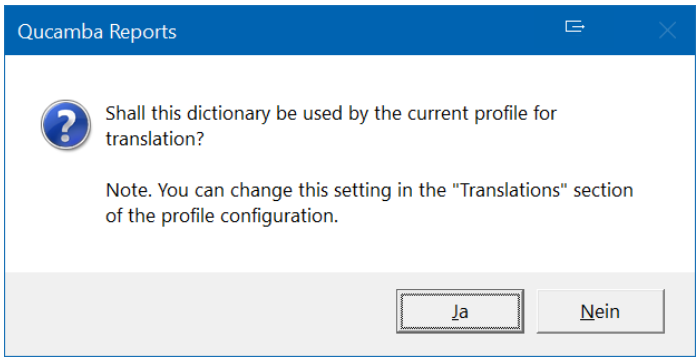
Once the dictionary is created successfully, you will be prompted to define the target languages for the dictionary.



All target languages of the currently edited app will be proposed. Select the desired target languages and confirm the dialog.

You may add or remove target languages at any time later via the dictionary editor.

If no dictionary has yet been assigned to the current profile, you now have the option to assign the new dictionary directly in the profile.



You can modify this selection at any time later and assign multiple dictionaries within the same profile.

4.2 Editing a Dictionary

When editing dictionaries, you can add or remove target languages and translate specific source texts (source language code: en-US) into the dictionary's target languages.

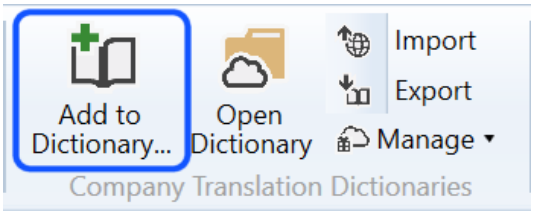
These translations can be entered manually or imported from the translation view.

Similar to the translation view, a full-text search is available here to help you browse dictionary entries.

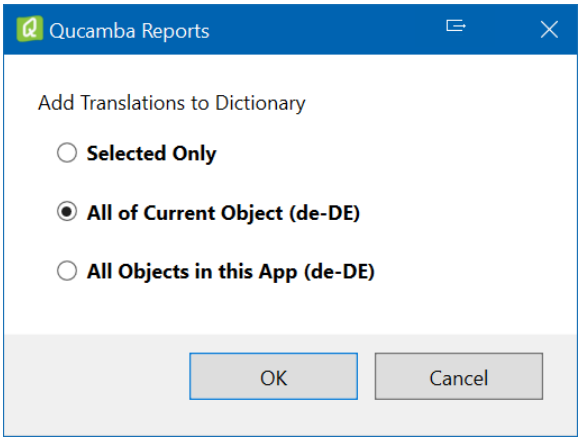
All changes made to dictionaries are immediately available to all users sharing the same Qucamba company license.

4.2.1 Adding Existing Translations to a Dictionary

Dictionaries can be populated with entries directly from the translation view. To do this, select the desired entries and click **Add to Dictionary...** in the **Company Translation Dictionaries** ribbon group.



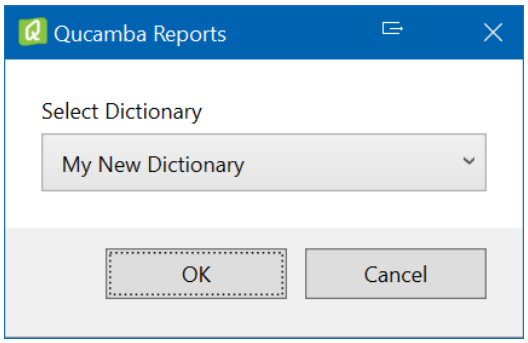
If you do not select multiple translations beforehand, a dialog will appear allowing you to choose which translations to add.



The selection of languages to be added is automatically based on the current language settings in the translation view. Therefore, make sure to enable or disable the desired languages via the language selector beforehand if necessary.

Select the translations you want to add to the dictionary and confirm the dialog.

In the list of dictionaries that now appears, select the dictionary to which you want to add the entries.



By default, the first dictionary configured in the profile's dictionary list is preselected.

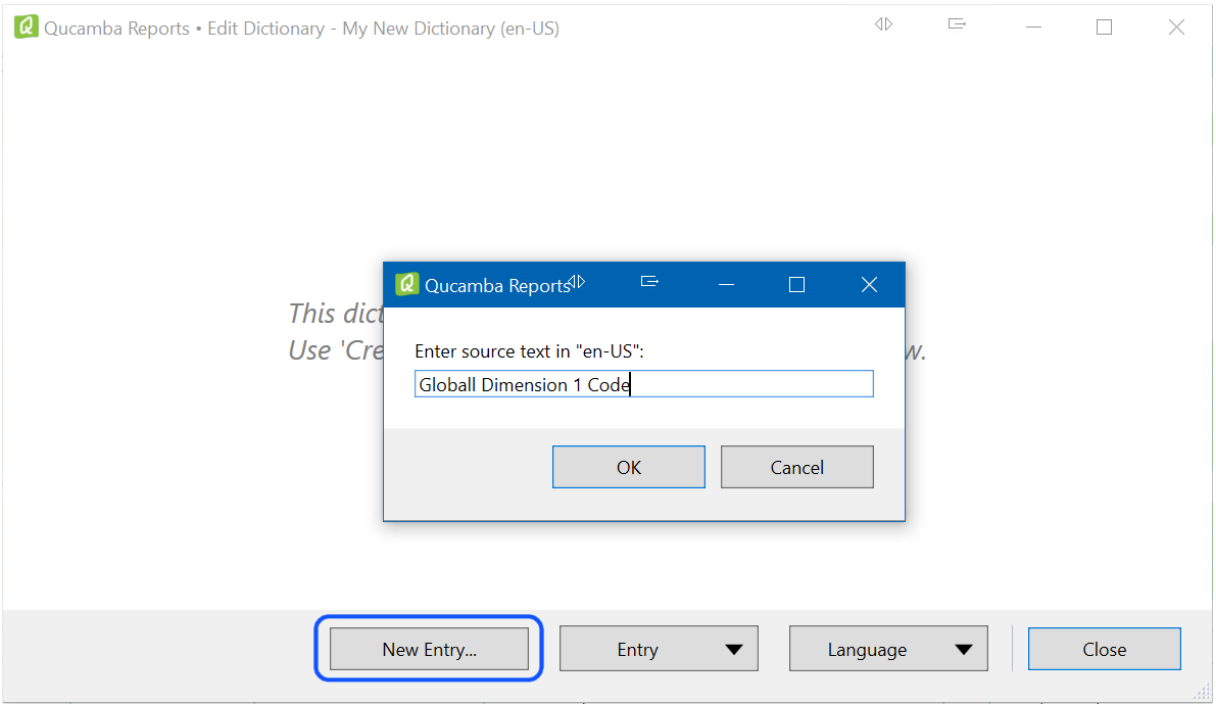
Once you confirm the dialog, the entries are added to the dictionary and are also available to other users with the same company license.

4.2.2 Editing a Dictionary in the Dictionary-Editor

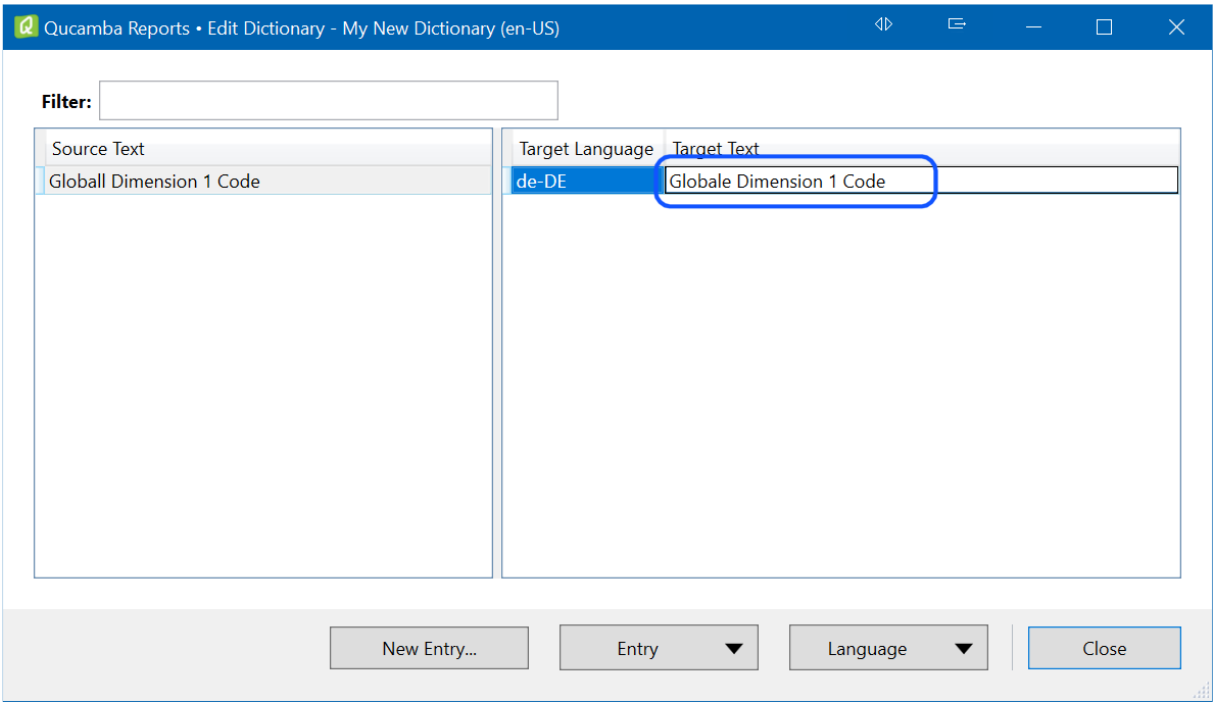
To edit a dictionary directly in the Dictionary Editor, click the **Open Dictionary** action in the **Company Translation Dictionaries** ribbon group.

Then select the dictionary you want to edit.

In the Dictionary Editor that now appears, you can add new source texts. To do so, click the **New Entry...** button and enter the source text (language code: en-US).

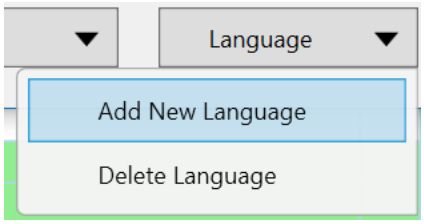


After confirming the dialog, the source text will appear on the left side; you can enter the translations into the dictionary's target languages on the right side.

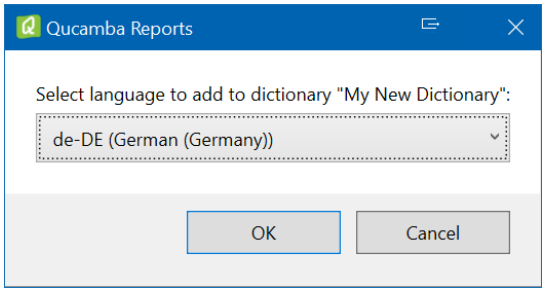


If you have not yet added any target languages to the dictionary, you can do so now. A dictionary can contain any number of target languages.

To add a new language, click the **Language** button and select **Add New Language**.



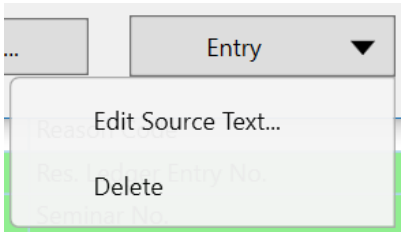
From the list of available languages, select the desired target language and click **OK**.



> **Tip**
>
> You can also enter the target language using the keyboard. The language selection will then jump directly to your input.

You can also remove existing target languages from your dictionary via the **Language** menu. Please note that this action will permanently delete all translations for the selected language from the dictionary and will also affect all other users sharing the same Qucamba company license.

If you want to edit the source text later or delete the entire entry including its translations, use the **Entry** button.



4.3 Importing and Exporting Dictionaries

Dictionary data can be imported and exported. For example, you can import existing translations from XLIFF files or Business Central apps into a dictionary, edit the content, and then export or use it for further translations.

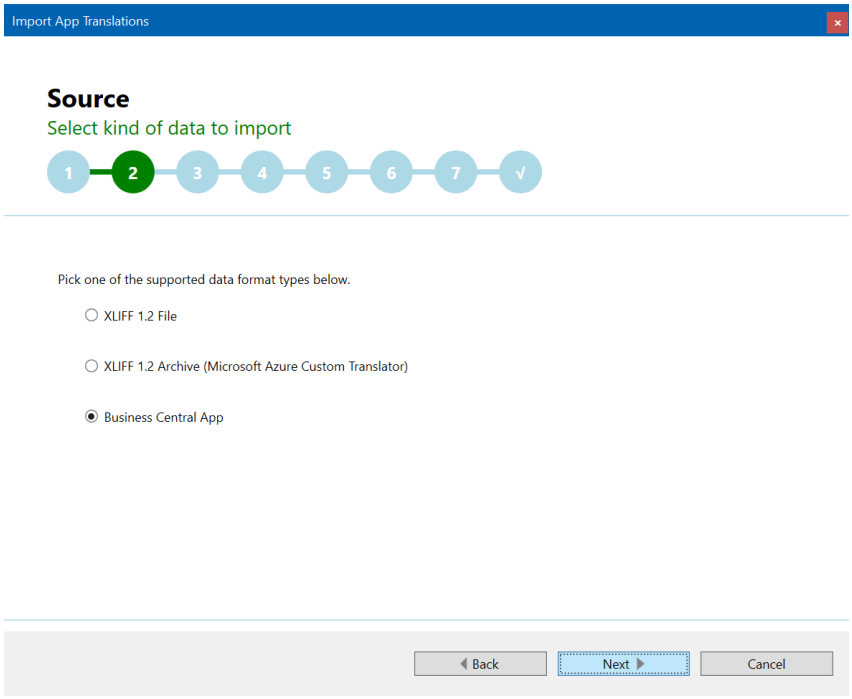
Exporting is useful, for example, when generating training material for Microsoft Azure Translator. The process of automatic translation using Microsoft Azure Translation Services is described in detail later in this document.

Both import and export processes are wizard-driven and guide you step by step through the task.

4.3.1 Dictionary Import

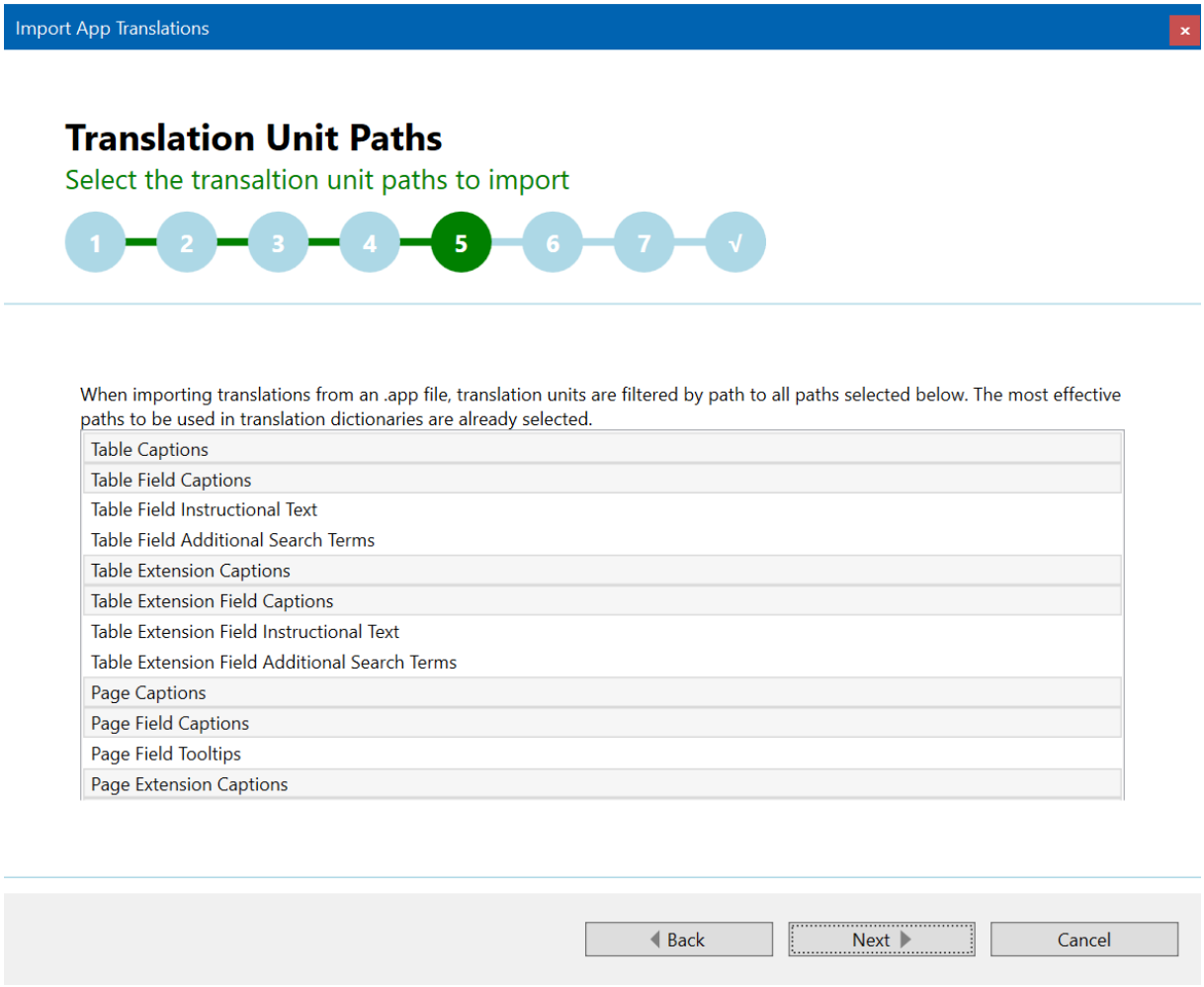
Before importing, ensure that the dictionary into which translations will be imported already exists. It may already contain data; existing translations may be updated during the import process.

During import, you can choose between an XLIFF file, a training data file for Microsoft Azure Translator, or a Business Central app.



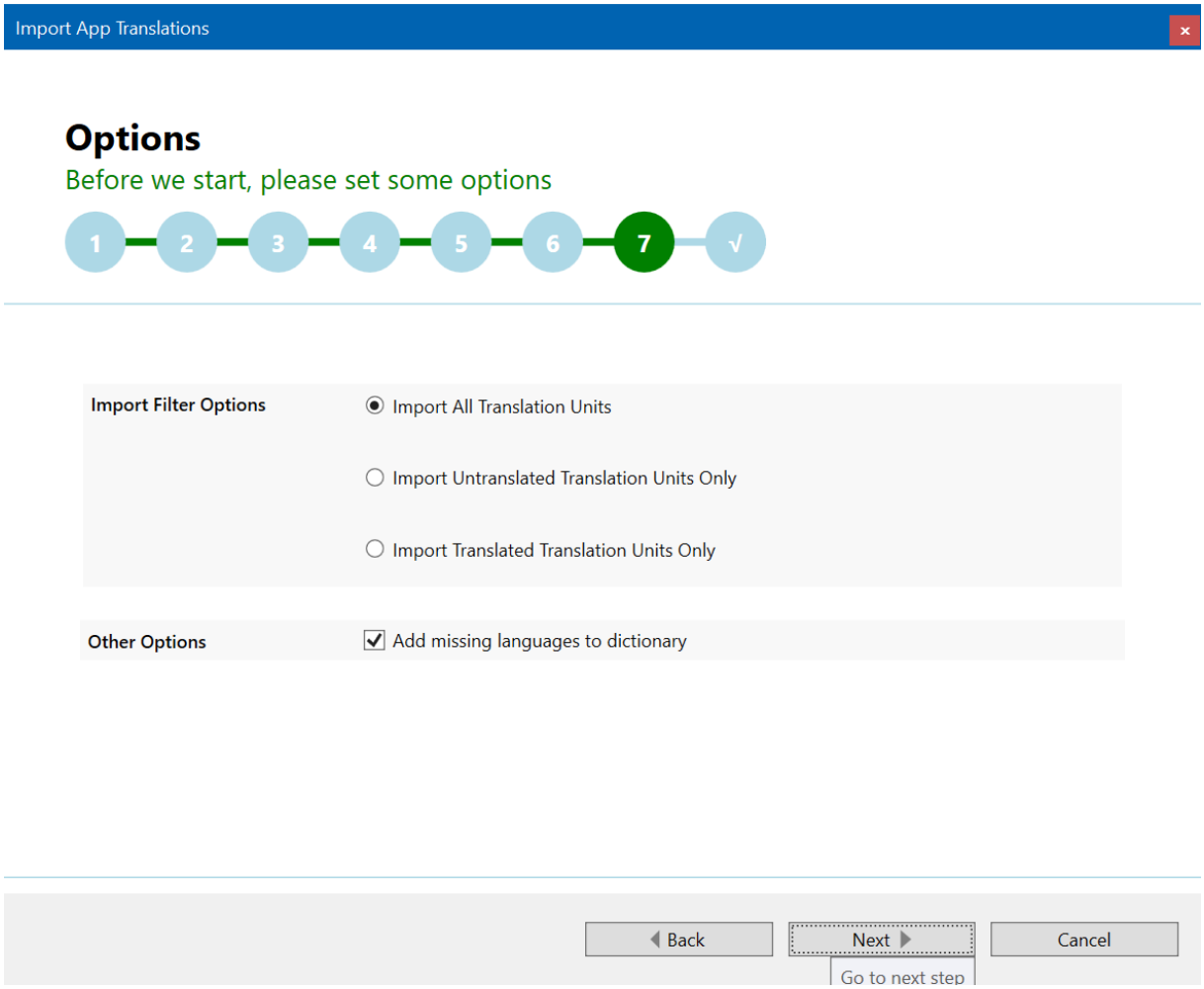
In the next step, select the file or app(s) from which translations are to be imported. When importing from apps, you will also be prompted to select the specific translation files.

The following wizard step allows you to choose which types of translations to import. Typically used types are preselected.



Next, select the dictionary into which the data should be imported. Please note that existing entries may be updated with imported content.

In the final wizard step, you can configure additional import options.

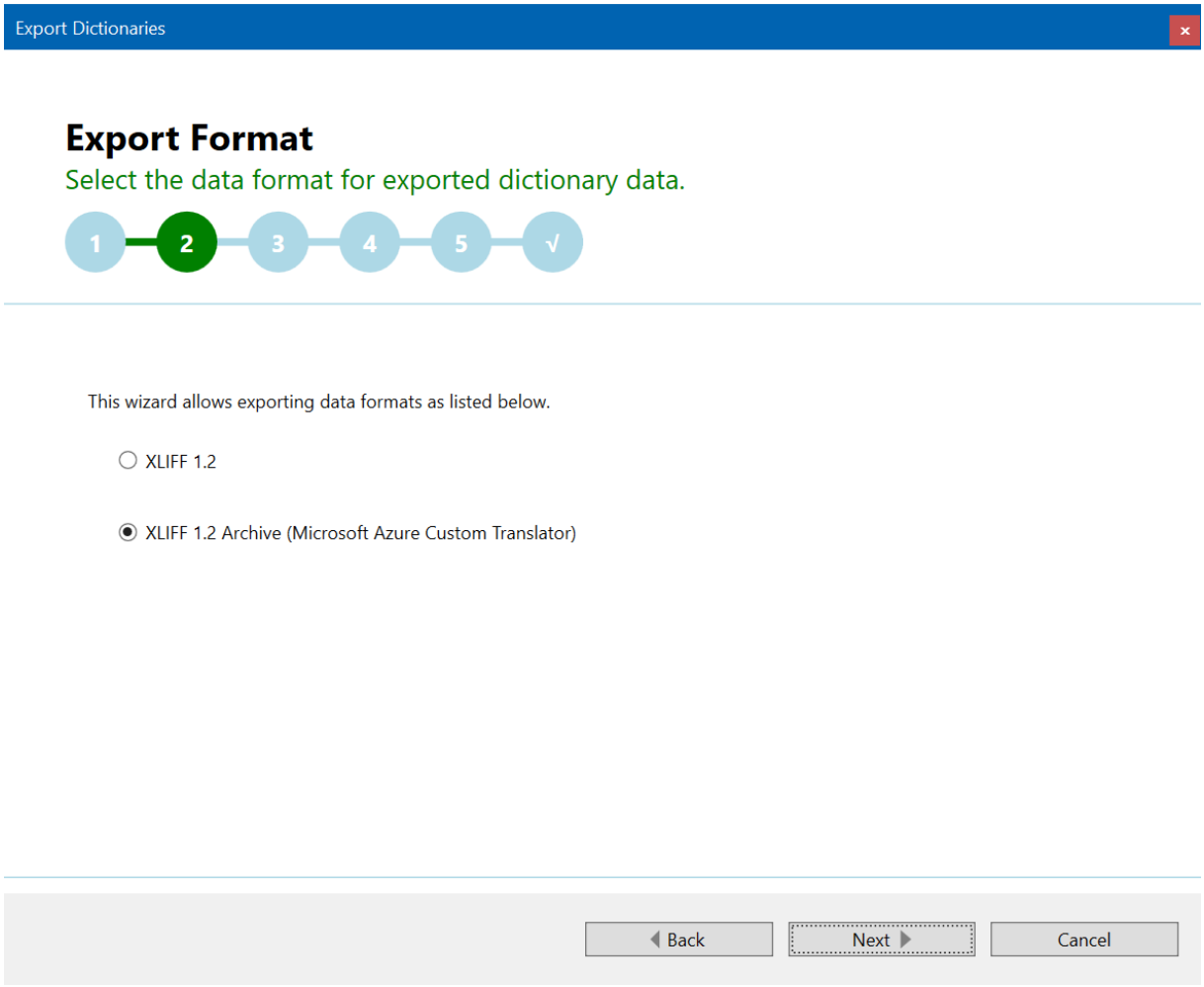


For example, you can choose whether to import only missing translations, only existing ones, or both.

Another option allows the wizard to automatically add missing target languages to the dictionary. If this option is not enabled, translations for unsupported languages will be skipped during the import.

4.3.2 Dictionary Export

First, select the export format for the export.



Use **"XLIFF 1.2 Archive"** if you want to generate training material for Microsoft Azure Translator. Detailed information on this can be found later in this document.

Next, select the dictionaries you want to export. Multi-selection is supported, allowing you to include multiple dictionaries in a single export.

You will then see a list of target languages contained in the selected dictionaries. Choose the target languages you want to export.

Finally, specify a file to which the export should be written.

4.4 Conclusion

Dictionaries play a central role in ensuring consistent, high-quality automated translations across all projects. By maintaining standardized terminology and enabling collaborative editing within the

organization, they significantly enhance both automated and manual translation processes. Features such as cloud-based access, priority control, import/export options, and integration with external services like Microsoft Azure Translator make dictionaries a powerful tool for professional translation workflows. When used effectively, they help streamline translation efforts, improve overall translation quality, reduce errors, and improve efficiency across teams.

5 Translation Methods

Beside manual translation, Qucamba Reports also facilitates automated translations. Automated translations can be of the following types:

1. Lookup terms in your own dictionaries, e.g. dictionaries that hold translations for special terminology of a particular branch
2. Find similar translations in dependency apps with interactive selection of translation if multiple matches are found
3. Use Microsoft Cognitive Services an external text translation service
4. Use Microsoft Custom Translator with self-trained models for increased translation quality

By default, Qucamba Reports applies a combination of these methods to increase the overall translation quality. The order in which available translation methods are applied is as follows.

1. Dictionary based translation
2. Translation based on existing translations
3. Automated translation using a self-trained model
4. Automated translation

While dictionary based translations result in highest quality on highest cost, quality decreases down these steps until automated translations are used resulting in lowest quality at lowest cost. However, when using well prepared dictionaries as well as well trained translation models quality of automated translation will meet requirements in most cases since it will mostly translate longer phrases like tooltips. While tooltips are usually not found as exact matches in dictionaries or in base app translations, they are easier to translate by a translation service due to their nature of usually containing entire sentences instead of just a few words.

5.1 Dictionary Based Translation

Qucamba Reports comes with an integrated dictionary functionality that allows users to create one or more dictionaries on their own. By default, these dictionaries are empty. To fill a dictionary with data, entries can be added manually or automated by extracting selected translations from existing translations e.g., of dependency apps or other Xliff content.

Users can work on the same dictionary at the same time. To ensure that users will not overwrite each other's entries, working with dictionaries requires a continuous online connection to the qucamba.com cloud service.

After creating dictionaries, each profile can be configured to use various dictionaries in a particular order. When using automated translations, Qucamba Reports first checks the profile's dictionaries in order of appearance before it starts using the other translation methods. This way, branch specific terms can be specified in a dictionary that overrides translations for the same terms from e.g. a base app.

5.2 Translation based on Existing Translations

By preferring translations that have been defined before, same source texts are always translated in the same way as they have been translated in base apps. In case there are different translations defined for the same source text, Qucamba Reports optionally lets the user select the most appropriate translation from a list.

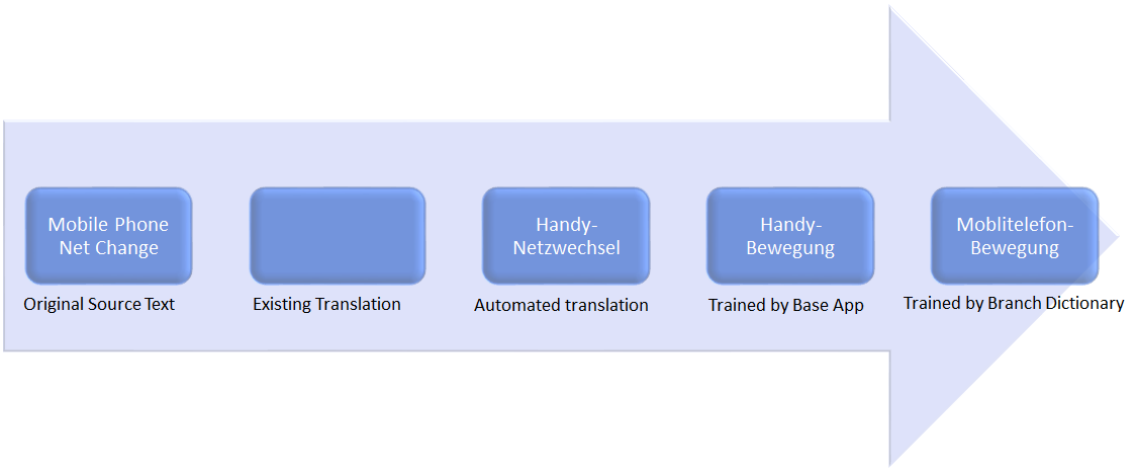
Since every app usually adds many new source texts that are not yet known by any dependency app at all, there will be many source texts remaining untranslated. The user can decide to edit these translations manually or configure the translation service for automated translations.

5.3 Automated Translation

Automated translations utilize Microsoft Cognitive Services Text Translator Resources. By clicking the „Auto Translate“ action in the Qucamba Reports Translate module, Qucamba Reports first applies a translation based on existing translations as described in the previous section. All remaining translations in all target languages are then translated externally by calling the external translation service. This approach results in translations of a certain level of quality. However, the external translation service doesn't know anything about the Business Central terminology or the terminology of a particular branch. This is when trained translation models come into power.

5.4 Self-Trained Model Translations

Of self-trained model translations one can think of as the successors of automated translations. The following illustration shows the differences between the translation methods mentioned above. In this example, the source text "Mobile Phone Net Change" which is a caption of a field that calculates the net change for a particular mobile phone is to be translated to German language.



Though the application contains translations for "Net Change" (German: "Bewegung"), the source text "Mobile Phone Net Change" cannot be found in any base app. Thus, target value of this translation unit remains empty.

In the next step, the external text translation service uses a basic text translation resource. The result shows what the "A" in "AI" stands for: the AI understands that we are talking about mobile phones and thus it not only decides to use the colloquial translation for mobile phone (i.e. "Handy") but also assumes that we are going to change the network provider of the mobile phone (German: "Netzwechsel").

When we use a custom translator that is trained by the base app, it starts to understand that "Net Change" actually means "Bewegung" in German but still it translates "Mobile Phone" to the colloquial phrase "Handy".

Finally, when we *additionally* train the custom translator with a dictionary that translates "Mobile Phone" to "Mobiltelefon", this results in the intended translation "Mobiltelefon-Bewegung".

Based on this knowledge, using a self-trained model increases the overall quality of translation tremendously. However, there will remain some inconsistency even when using a self-trained model.

The following table illustrates an example.

En-US, English (USA)	Translation to de-DE, German (Germany)	
Example Source Text	**Correct translation**	**Self-trained Model**
No. of Repairs	Anzahl Reparaturen	Nein. Von Reparaturen

We should now understand that the process of training a model is a repetitively refining task. In this example, we would add "No. of" to our branch dictionary and translate it to German: "Anzahl".

Training a model can be based on various kinds of training material, i.e.

- Language aligned training documents
- Phrase dictionaries
- Sentence dictionaries

As a thumb of a rule, the more data is used to train a model the better the quality of translation becomes. Best results can be achieved by using aligned training documents. However, the Microsoft Custom Translator requires to upload at least 10.000 sentences per pair of documents.

At Qucamba, we tested various approaches and received the most effective results (i.e. best results at lowest cost) by training a model using a phrase dictionary with data taken from a custom dictionary filled with

a) Table captions, table field captions etc.
b) branch specific terms

The dictionary data can then be uploaded to the Microsoft Custom Translator e.g. as Xliff file.

Note, that due to format issues it is not possible to upload the original Xliff files from the Business Central base app to Microsoft Custom Translator. Any attempt to do so will fail during processing of the Xliff data.

6 SETTING UP AUTOMATED TRANSLATIONS IN QUCAMBA REPORTS

Qucamba Reports supports all three automated translation methods mentioned above. We tried to make it as easy as possible for our customers to configure the automated translation service.

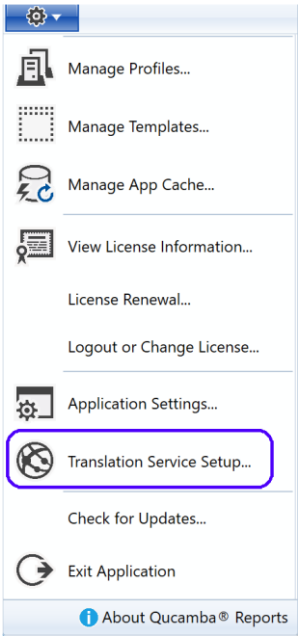
For example, as soon as you configure an external translation service, this configuration is distributed to all installations of your Qucamba Reports license by the integrated cloud replication mechanism. As soon as one employee successfully configured the translation service, all other employees will be able to use automated translations as well. The same applies for self-trained translation models.

6.1 SETTING UP TRANSLATION BASED ON EXISTING TRANSLATIONS

For translation based on existing translations, everything you need to do is open the TRANSLATE section in Qucamba Reports *and wait until the progress indicator besides the app selection control disappears*. Then, the "Suggest Translation" action is used to search for similar source texts in all base apps. In case that multiple translations for the same source text are available, the most appropriate translation can be picked from a list.

6.2 SETTING UP AUTOMATED TRANSLATIONS

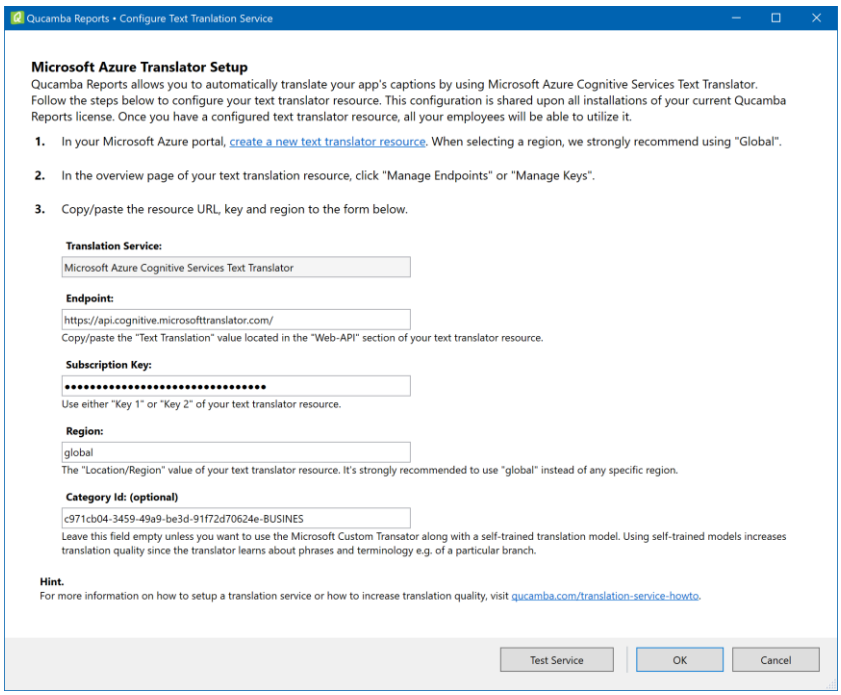
To setup automated translations, select „Translation Service Setup…" located in the blue application settings menu button in the top left corner of the Qucamba Reports application window.



As soon as you have signed-in to Microsoft Azure, the steps of configuring the translation service are very straight forward:

a) Create a text translation resource
b) Locate the resource configuration data
c) Copy/paste the endpoint, key and region of the resource to the configuration screen

The following illustration shows the configuration window in Qucamba Reports.

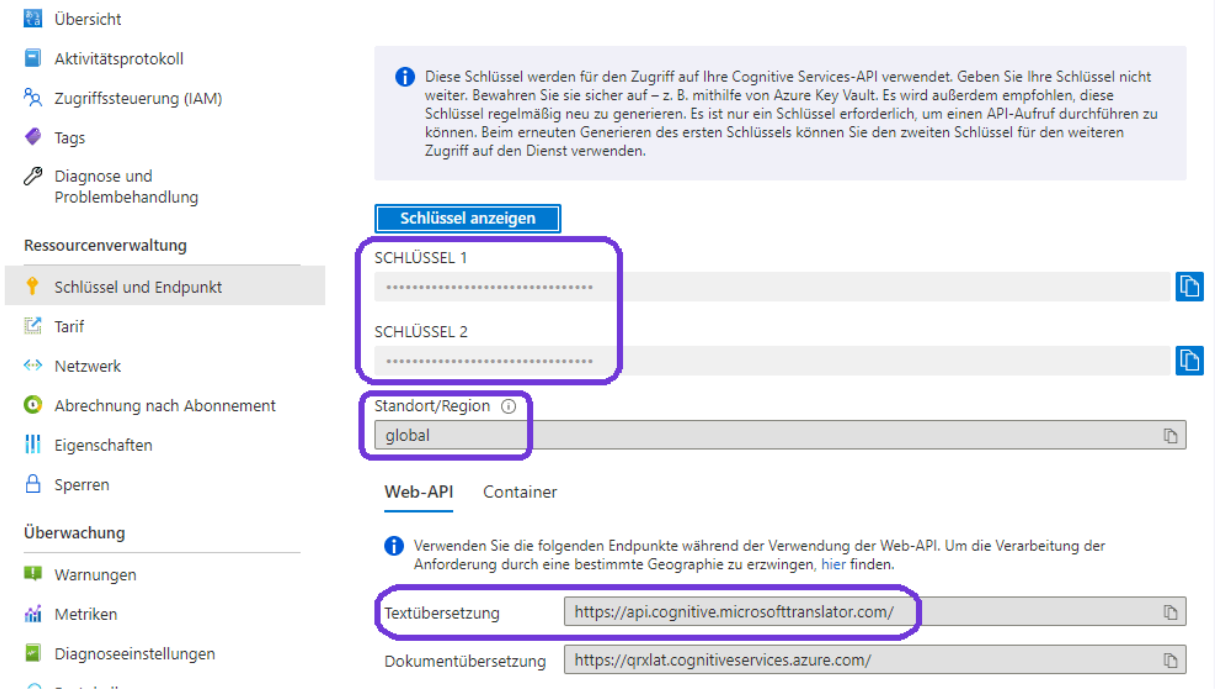


To create a new text translator resource, simply click the link located in step 1 of the configuration window. The link opens your browser and guides you directly to the creation of the resource.

When creating the text translator resource it is important to select „Global“ from the „Region“ selection list. Though it should be possible to specify a specific region, we were not able to authenticate against the translation service when using a region any other than „global“. However, since this might be corrected in the future, the region can be customized in the configuration dialog.

After the create request has been processed by Microsoft Azure, locate your resource and open ist overview page. With this overview page, click the „Show Endpoints“ or the „Show Keys“ link to collect the information required to configure the service. In particular, the information required is

1. The URL of the translator's endpoint which can be found in the WebAPI section of the resource overview
2. The resource region (as mentioned before, this is usually „global“)
3. One of the two available keys.

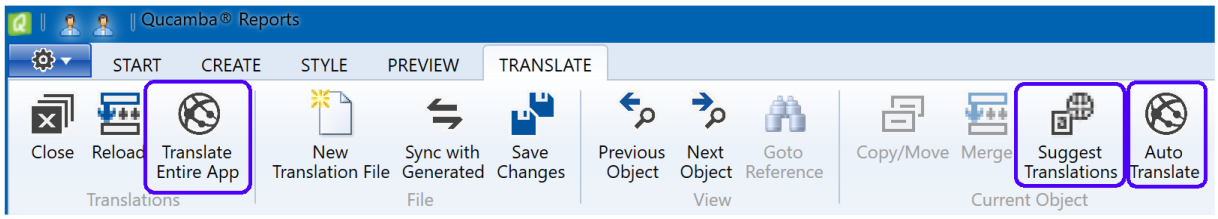


To the right of the fields you can find a „copy to clipboard"-button that copies the value to your Windows clipboard. Then, go to the corresponding input field in the Qucamba Reports Translation Service Setup window and paste the value from the clipboard.

For the moment, leave the „Category" field empty. This field is not required unless you are going to use a self-trained model which is described in detail below.

Finally, click the „Test Service" button to test you configuration. Please note, that it might take some time (usually up to a few minutes) until your text translation resource becomes available.

After configuring the text translation service, click „Auto Translate" in the „Current Object" section of the TRANSLATE module or „Translate Entire App" in the „Translations" section.



The following table explains the difference between the three actions marked in the illustration above.

Action	Description
Suggest Translations	For the *current object*, suggest translations based on existing translations of dependency apps. If multiple matches are found, the most appropriate translation can be picked from a list.
Auto Translate	For the *current object*, complete all missing translations by using the external ltext translator. Usually, you should first perform a translation suggestion based on existing translations before using the translation service.
Translate Entire App	This action allows you to select the languages to translates and performs an integrated translation by first consulting the dependency app for matching existing translations. If multiple possible matches occur, the most approriate translation can be picked from a list for each translation unit. Then the text translation service is called for all remaining translations.
New Translation File	When adding a new target language, one can not only let Qucamba Reports generate the corresponding translation file but also initiate an automated text translation.

6.3 SETTING UP SELF-TRAINED TRANSLATION MODELS

Self-trained models result in the highest quality that can be achieved when using automated text translation. For setting up a self-trained model, the following steps need to be accomplished.

1. In your Azure portal, create a *paid* text translation resource (at least S1)
2. Sign-up to Microsoft Custom Translator
3. Create a new workspace
4. Create a new project
5. Create and upload training material
6. Create a model based on the processed training material
7. Deploy a model in selected regions
8. Copy the project's „Configuration Id" to the Qucamba Reports Text Translation Service Configuration

Though on first sight, this seems like a difficult task it is quite easy to create and use trained translation models. The benefit is that the translator learns to know about the specific language of Business Central in general and of your branch in particular gaining a higher quality of translations.

The following sections describe the steps of creating a self-trained translation model in detail.

6.3.1 Create a paid text translation resource

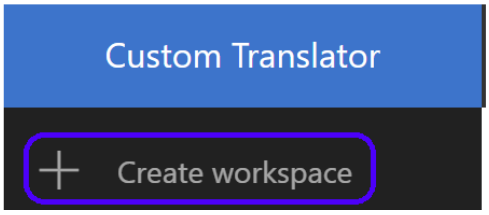
Deploying a trained model, is restricted to the use of a paid text translation resource, which means that instead of an F0 subscription you are required to have at least an S1 text translation resource created. The process of creating a paid text resource is the same as explained above but one of the paid pricing models must be selected. In case you already started to configure the Microsoft Custom Translator with a free text translator resource, you can change the resource key in the „Settings" section of the Microsoft Custom Translator portal. Remember that if you change the key, you will also need to update the key in the Qucamba Reports Translation Service Setup window.

6.3.2 Sign-up to Microsoft Custom Translator

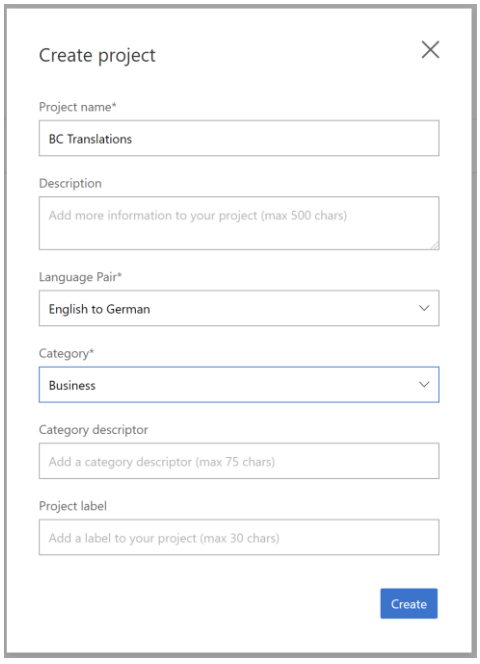
First time you go to https://portal.customtranslator.azure.ai/ you are automatically asked to sign up to the Microsoft Custom Translator.

6.3.3 Create a new workspace

In the top left corner of the Microsoft Customer Translator portal, click on „+ Workspace" as illustrated below.

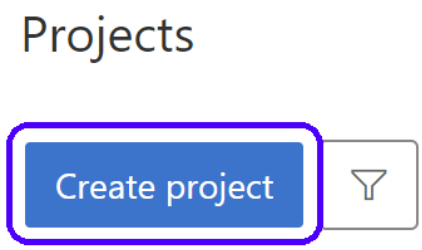


Enter a name for the workspace and pick a language pair as well as a category from the list. In the illustration below we are going to create a workspace for translations from English to German language.

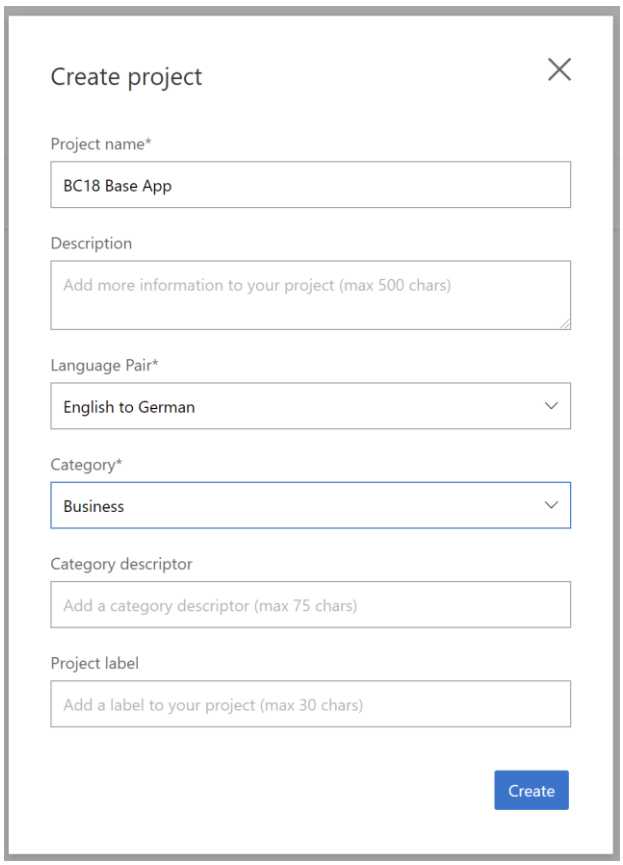


6.3.4 Create a new project

Translation models and training material are stored in a project within the workspace. To create a new project, select „Projects“ in the navigation bar and click „Create project“ as illustrated below,

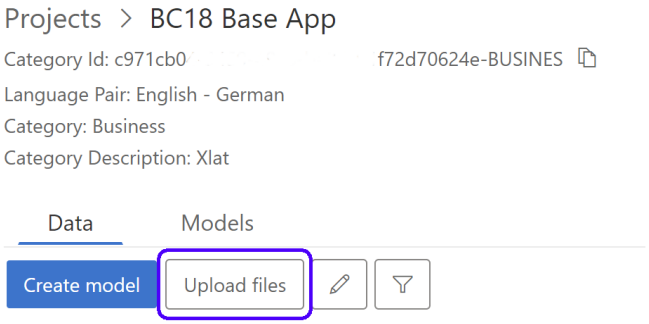


Again, enter the appropriate information in the project creation form as illustrated below and click „Create“.



6.3.5 Create and upload training material

To upload training material, open the project by clicking the project name in the list of projects. Then, click „Upload Files“.



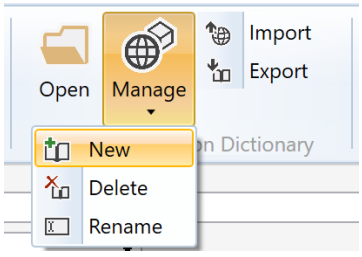
There are various document types available for training the model, e.g. dictionaries of phrases, dictionaries of sentences, training, tuning and testing. The latter one should not be used for training the model but for testing translation quality after a training.

Various approaches are available for training a translation model. In this example, we are going to use a phrase dictionary based on an Xliff file. The idea is to increase quality of translation in the first step by teaching the model the basic terminology of a Business Central application. One way to achieve this would be to get an Xliff file from the base app and upload it to Microsoft Customer Translator. However, these Xliff files are not well-formed resulting in the training process to break. Moreover, these files contain a lot of data with many duplicates.

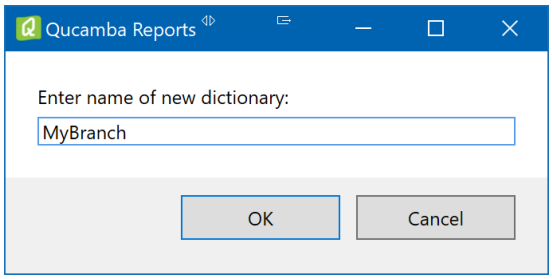
For this reason, Qucamba Reports contains editable dictionaries that can be fed manually or by importing translation units from apps. The nature of these dictionaries is that each source entry is unique and can contain any number of target languages with their corresponding translations.

Each dictionary is stored in the Qucamba Cloud. Though usually, Qucamba Reports uses replication mechanisms to allow working offline most of the time, working with dictionaries requires a continuously available online connection to qucamba.com. This allows multiple users of the same Qucamba Reports to work on same dictionaries at the same time.

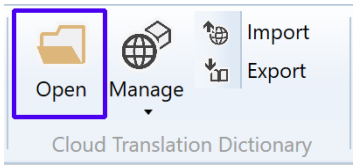
When working with dictionaries the first time, at least one dictionary needs to be created. To create a new dictionary, select the TRANSLATE ribbon tab and in the “Cloud Translation Dictionary” group, click “Manage” and then “New”.



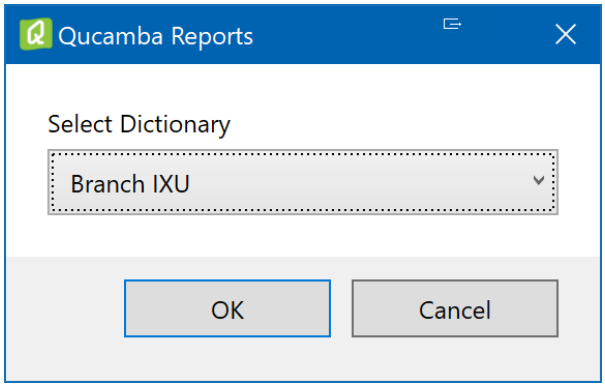
Enter a name for the dictionary and click OK:



To open an existing dictionary, click "Open" in the "Cloud Translation Dictionary" ribbon group.

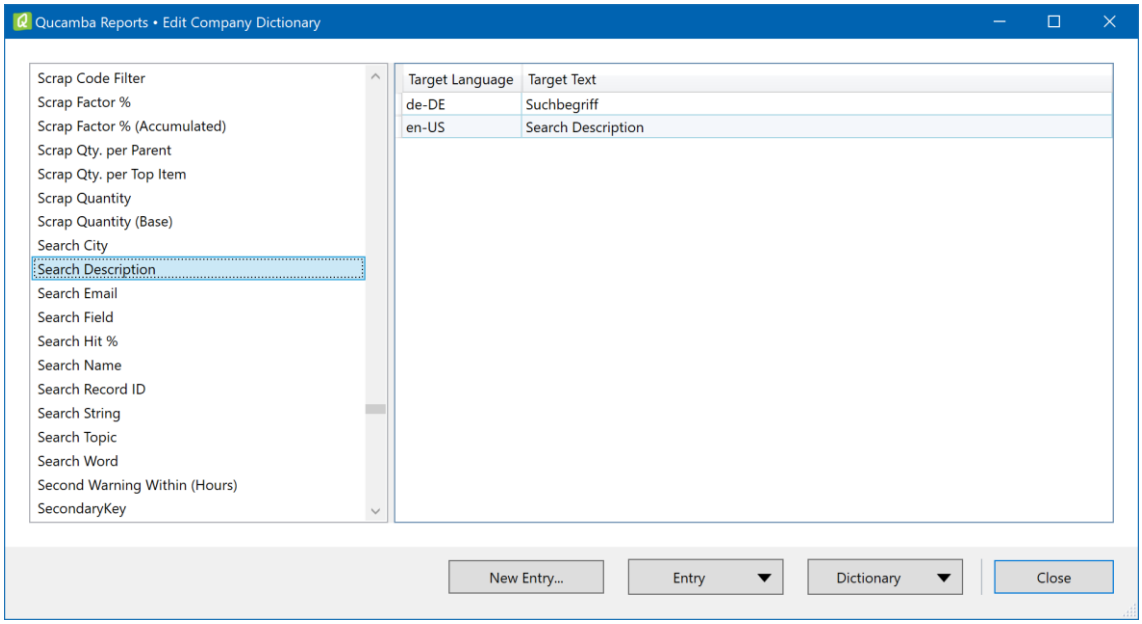


As long as there's only one dictionary created yet, Qucamba Reports instantly opens this single dictionary. However, if you have created more than one dictionaries, Qucamba Reports displays a list of available dictionaries and lets you choose the dictionary to work on as illustrated below. In this case, pick the appropriate dictionary from the drop down list and click OK.



When opening the dictionary, Qucamba Reports needs to load the dictionary content from the cloud. Depending on the amount and size of entries, this might take some time.

The illustration below shows the main window of the dictionary.

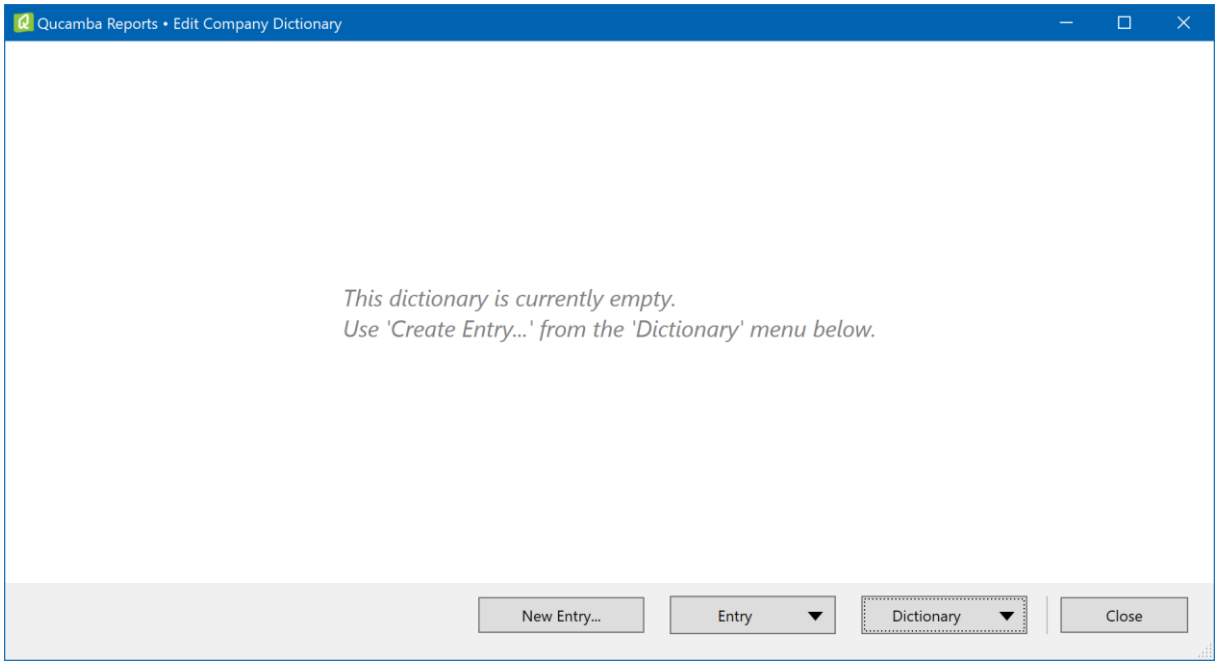


When importing translation units from an app, the dictionary allows to restrict the import to particular translation unit paths, e.g. to include all table captions and all table field captions only. The translation unit paths to import can be selected by the user during the import process.

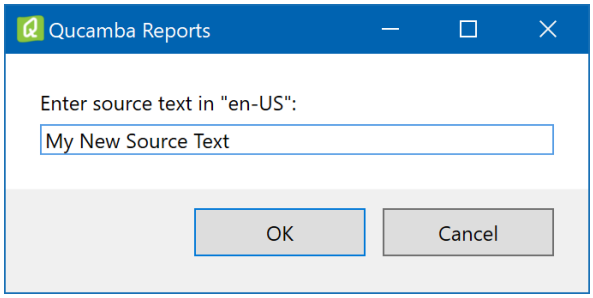
After importing translation units, translations can be edited and more translations like branch-specific phrases can be added to the dictionary manually. Finally, the dictionary can be exported to an Xliff file. Unlike the base app's Xliff file, this Xliff file is smaller in size, contains only the most relevant phrases, doesn't contain duplicates and it is well-formed so that Microsoft Custom Translator can process it successfully.

The dictionary is available in the "Translation Model" group located in the Qucamba Reports TRANSLATE section.

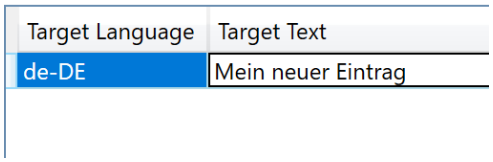
When you open the dictionary, it is still empty. However, Qucamba Reports already added the target languages that are defined in your app. You can add additional languages or remove existing languages at any time by selecting "Add language..." or "Remove language..." from within the "Dictionary" menu button.



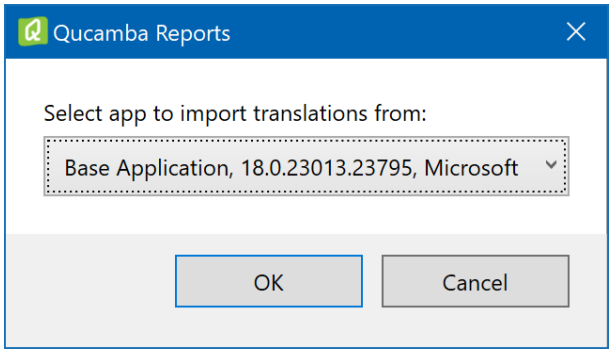
To add new dictionary entries manually, click “New Entry…” and enter the source text of your entry.



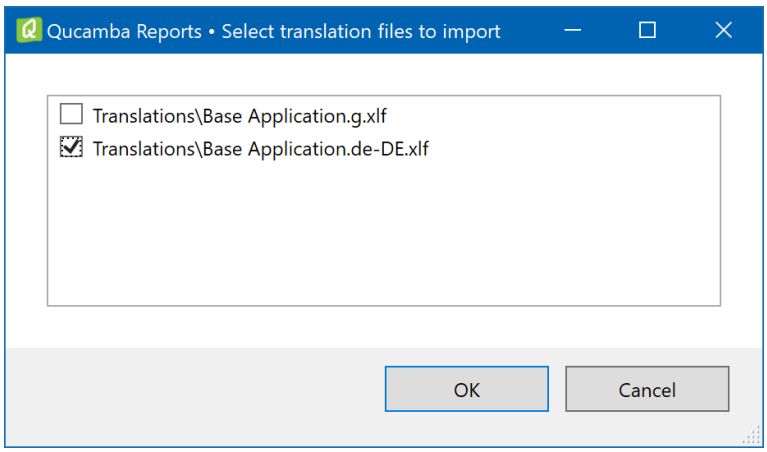
After clicking “OK”, the entry is added to the dictionary and automatically selected. The translation target values can now be entered in the right pane for each registered target language.



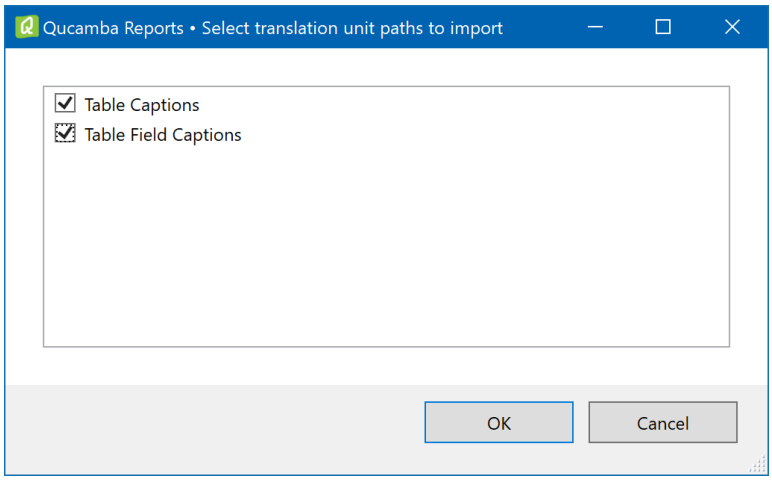
Usually, you start by adding translations from an app. From the “Dictionary” menu button, select “Import App” and in the app selection dialog, pick the Base Application from the list.



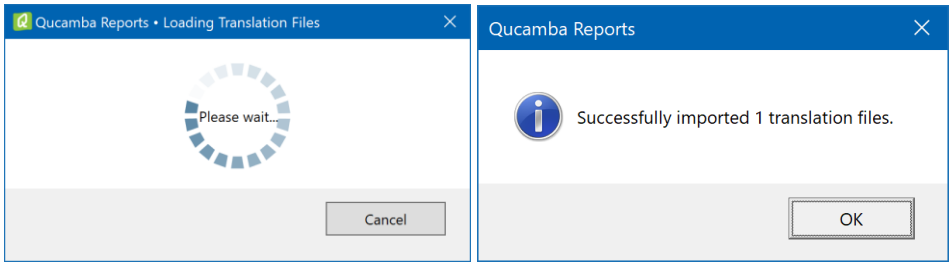
The following dialog presents a list of all translation files contained in the selected app. Place a checkmark in all languages to import into the dictionary and click "OK".



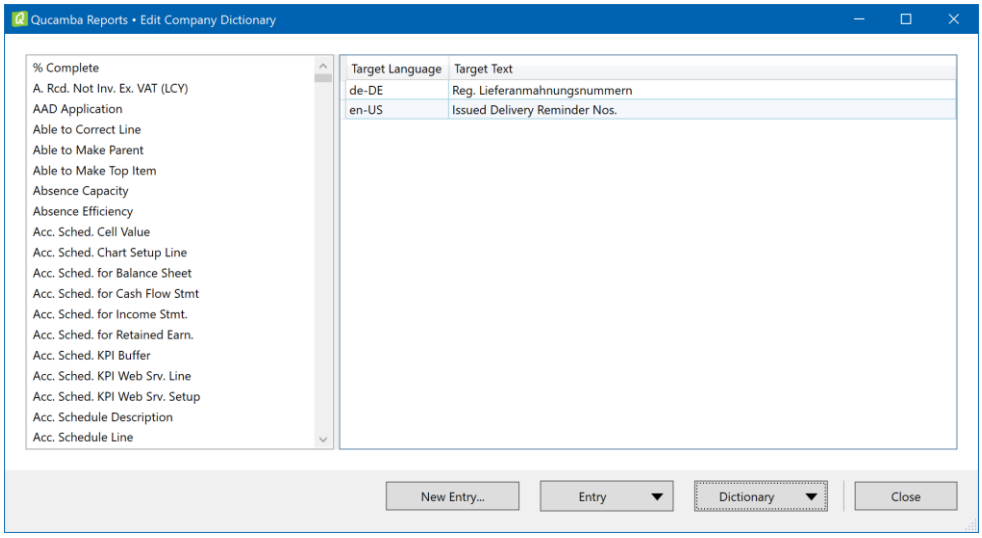
Next, place a checkmark near the translation unit paths you want to import and click "OK".



Wait until Qucamba Reports imported all translations into the dictionary.

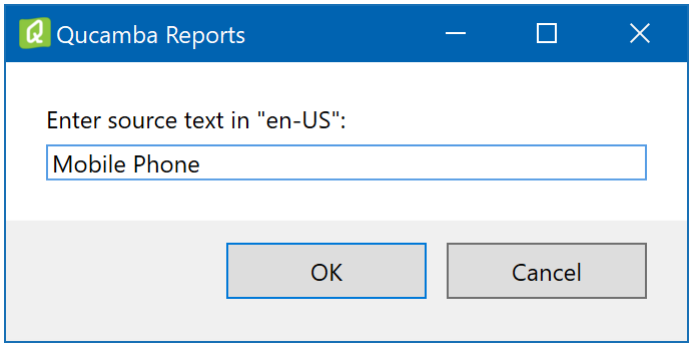


The dictionary has now been popluated as illustrated below.

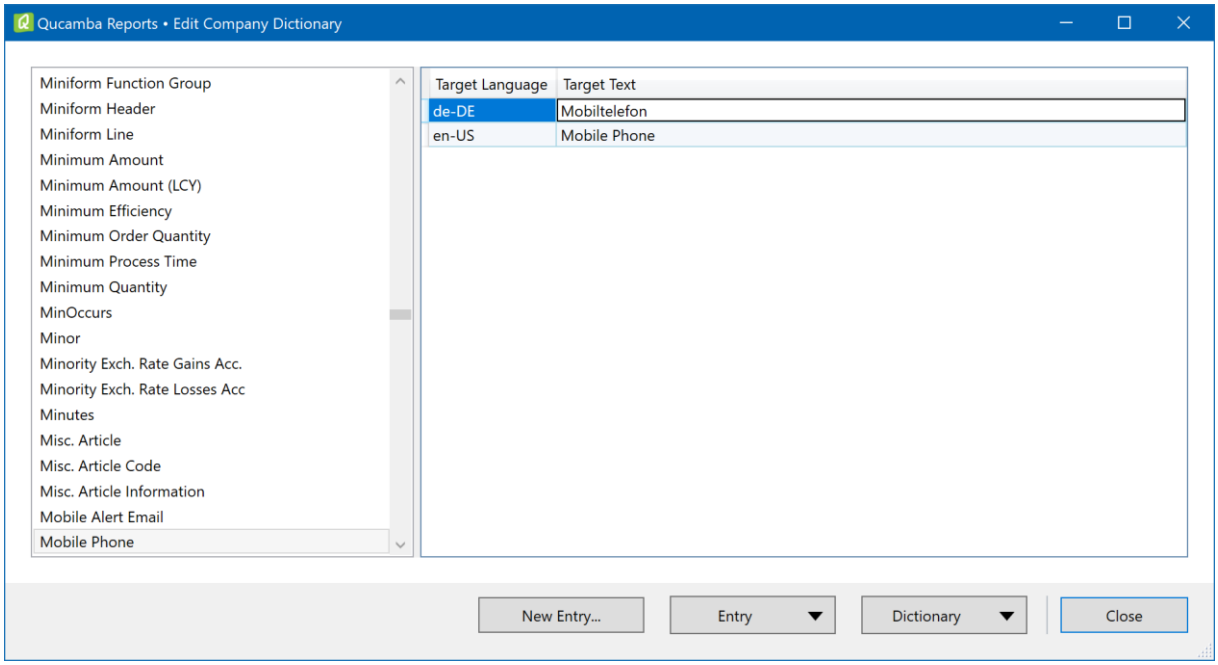


You might now want to continue adding custom phrases, e.g. to tell the translator that the most appropriate german translation of “Mobile Phone” is not “Handy” but “Mobiltelefon”.

Click the “New Entry…” button and enter the source text as illustrated below.

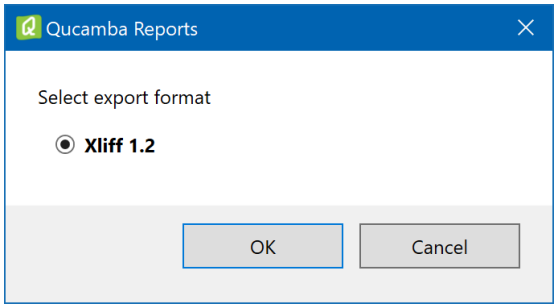


Then, enter the appropriate translations in the right pane.

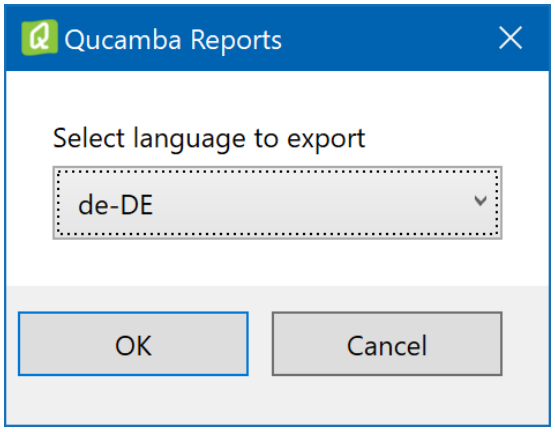


After adding all training data, select "Export Dictionary..." from the "Dictionary" menu button.

Select the file format and click "OK".

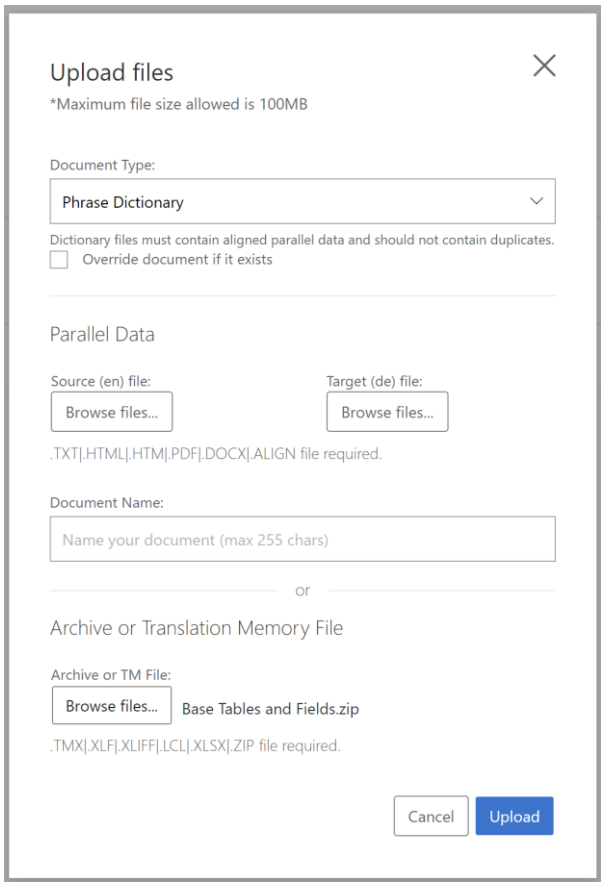


Then, select the language to export and click "OK".

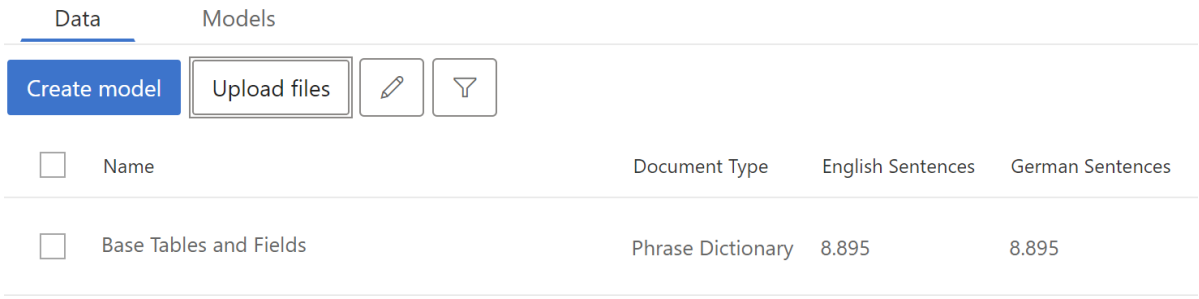


Finally, select a file and location and click "Save".

Before you can upload this file to Microsoft Custom Translator, you need to pack it into a ZIP archive file. Then in the translator portal, click on "Upload file" and fill out the form as illustrated below.



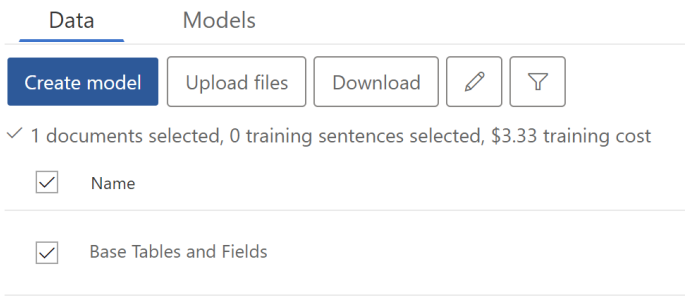
In this example, we leave the "Parallel Data" section empty and use the "Archive or Translation Memory File" section instead. Click "Upload" and wait until Microsoft Custom Translator successfully completed processing of your file.



You might want to check the content of the file by clicking on the file name.

6.3.6 Create a model based on the processed training material

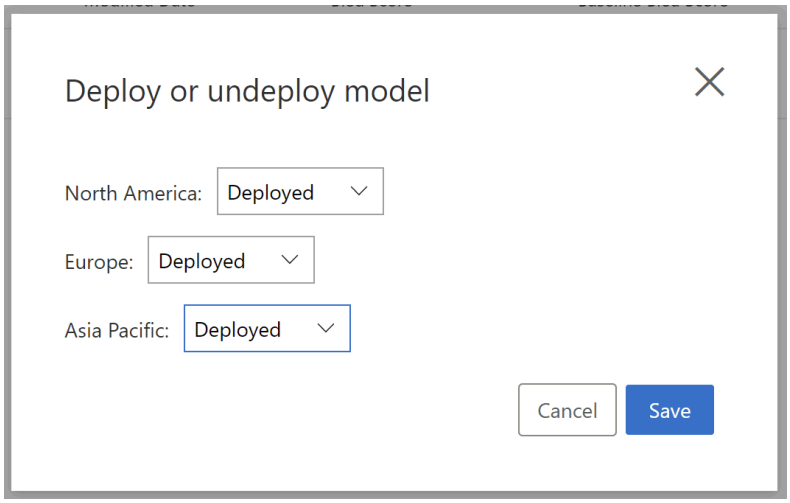
In Microsoft Custom Translator portal, open you project, select your training file(s) and click the "Create model" button.



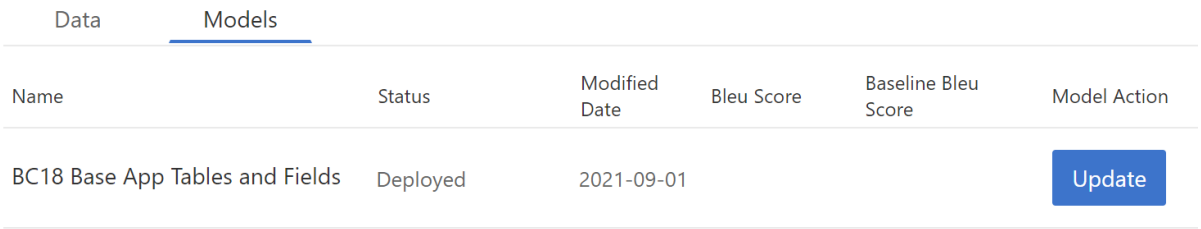
Carefully inspect the training cost before creating the model. The process of creation may take a while and after creation the model's status should show up as something like "Training succeeded".

6.3.7 Deploy a model in selected regions

Finally, you need to deploy your model by clicking the "Deploy" button. Select the regions in which the model should become available as illustrated below and click "Save"..

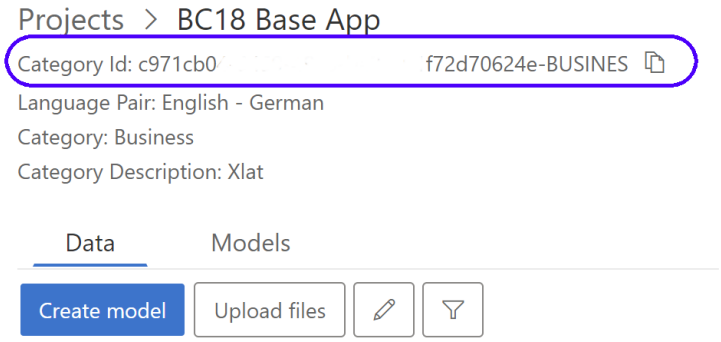


Again, the deployment takes some time and after successful deployment, the model entry should look like illustrated below.

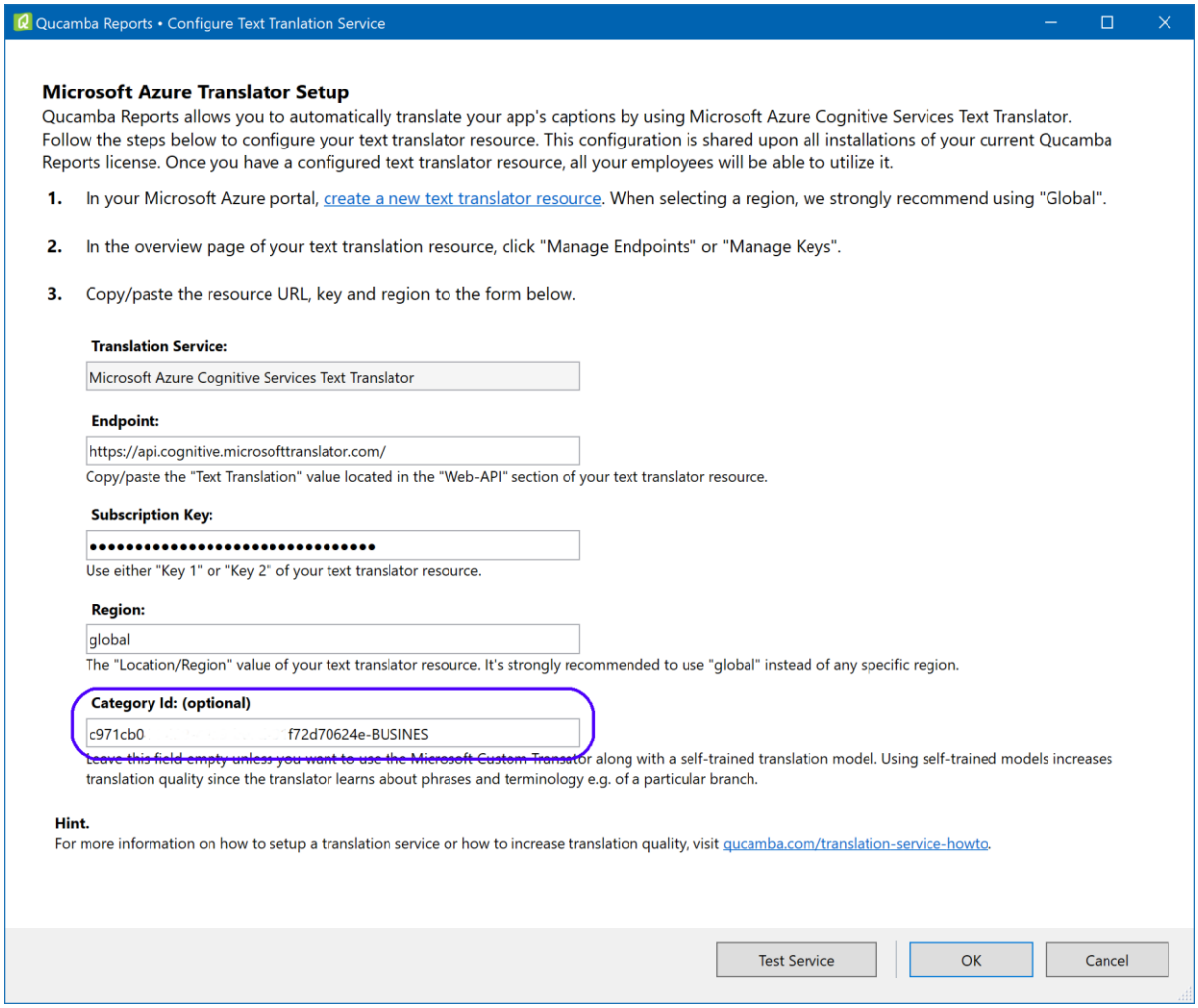


6.3.8 Configure Qucamba Reports to use a trained model

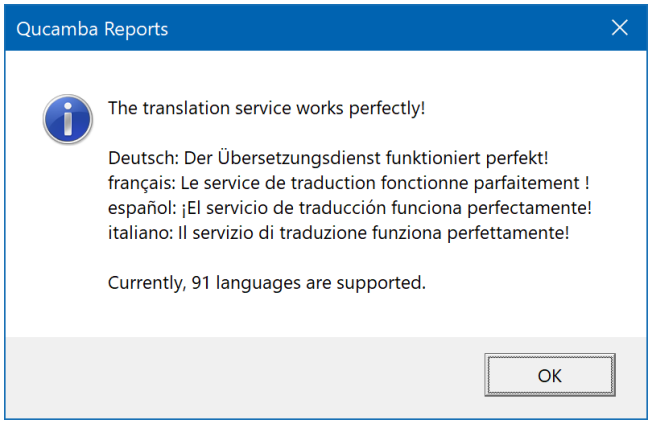
You are now ready to configure Qucamba Reports to use your trained model. Assuming that you already configured automated translations in the Qucamba Reports Translation Service Setup, the only step to take is filling in the "Category Id" of your model. You can find this id in the Microsoft Custom Translator portal as partially illustrated below.



Please note, that the category (in this case "BUSINES") is part of the id and should be copied as well. To copy the Category Id, click the button to the right of the id and paste it into the Qucamba Reports Translation Service Setup window.



Finally, click "Test Service" to check if the model works correctly.



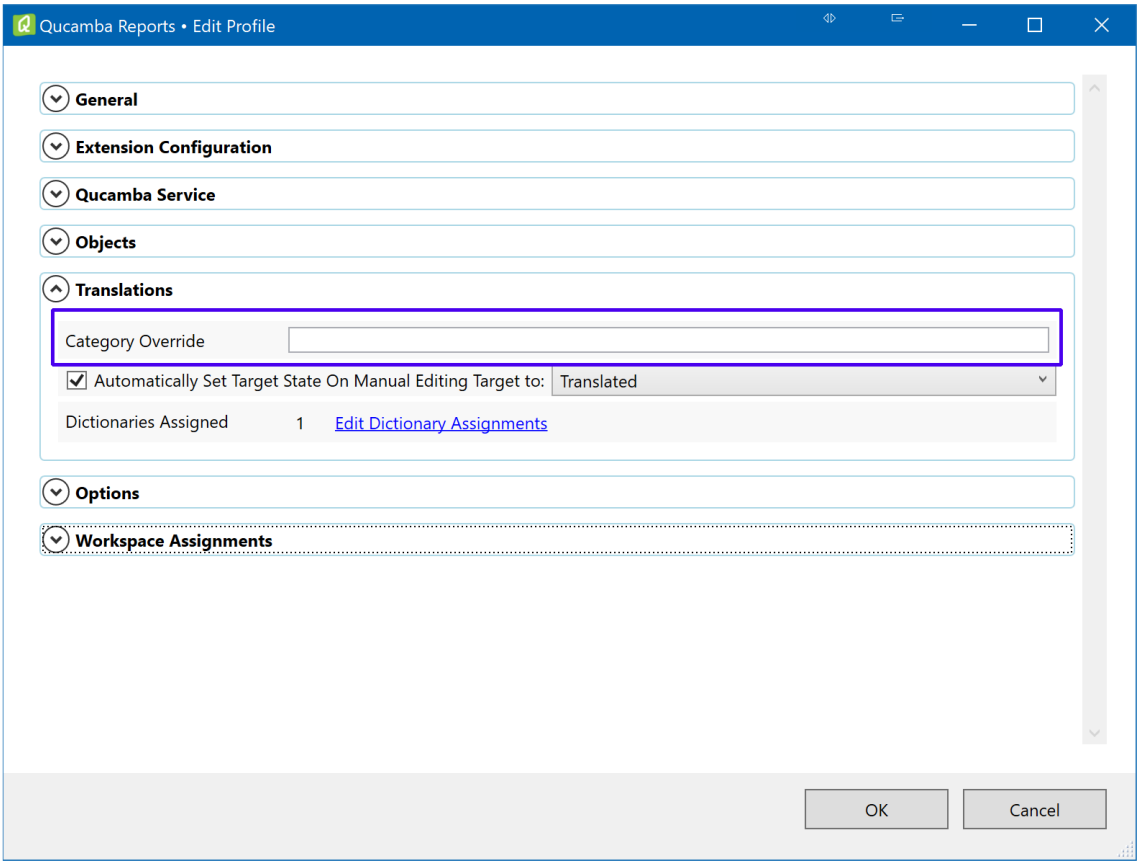
Please note, that it might take some time until your deployed model becomes available.

Each time you use the translation service now your trained model is applied, and translations should be of much better quality.

6.4 USING DIFFERENT SELF-TRAINED MODELS FOR DIFFERENT BRANCHES

In case you are an ISV that delivers various solutions to various markets and branches, you might want to use different self-trained models for each of your apps. Qucamba Reports also allows to configure the text translation service category id within the profile. If the profile has a configured category id, this setting overrides the category id in the text translation setup window. However, if a profile should use the category from the text translation setup window you should leave the profile's category id empty.

The illustration below shows the category id field from the profile details window.



To use a different text translation category id for this profile, just copy and paste the id in this field in the same way as described above for the text translation setup window.

7 BEST PRACTICES

In the previous sections, you learned how to configure Qucamba Reports for automated translations. In this section, we are going to explain best practices on how to translate an app regardless of whether you are going to translate it object-by-object or the entire app in one go and regardless of whether you are doing the initial translation or subsequent translations after editing your app.

7.1 GENERAL

There are various approaches of how to translate an app. Qucamba recommends a combination of translation methods in a particular order.

1. Translate by using one or more Translation Dictionaries,
2. Translate by using existing translations from base apps,
3. Resolve all remaining translations by using a *trained* online translator model.

After setting up translations in Qucamba Reports, you should start by creating two dictionaries.

a) First, create as "Base Translations" dictionary in which you import all translations of the base app.
b) Then, create one dictionary "Branch XYZ Translations" for each branch you are working on.
c) Enter the most important words and phrases into the branch dictionary. You may refine this dictionary later from time to time.

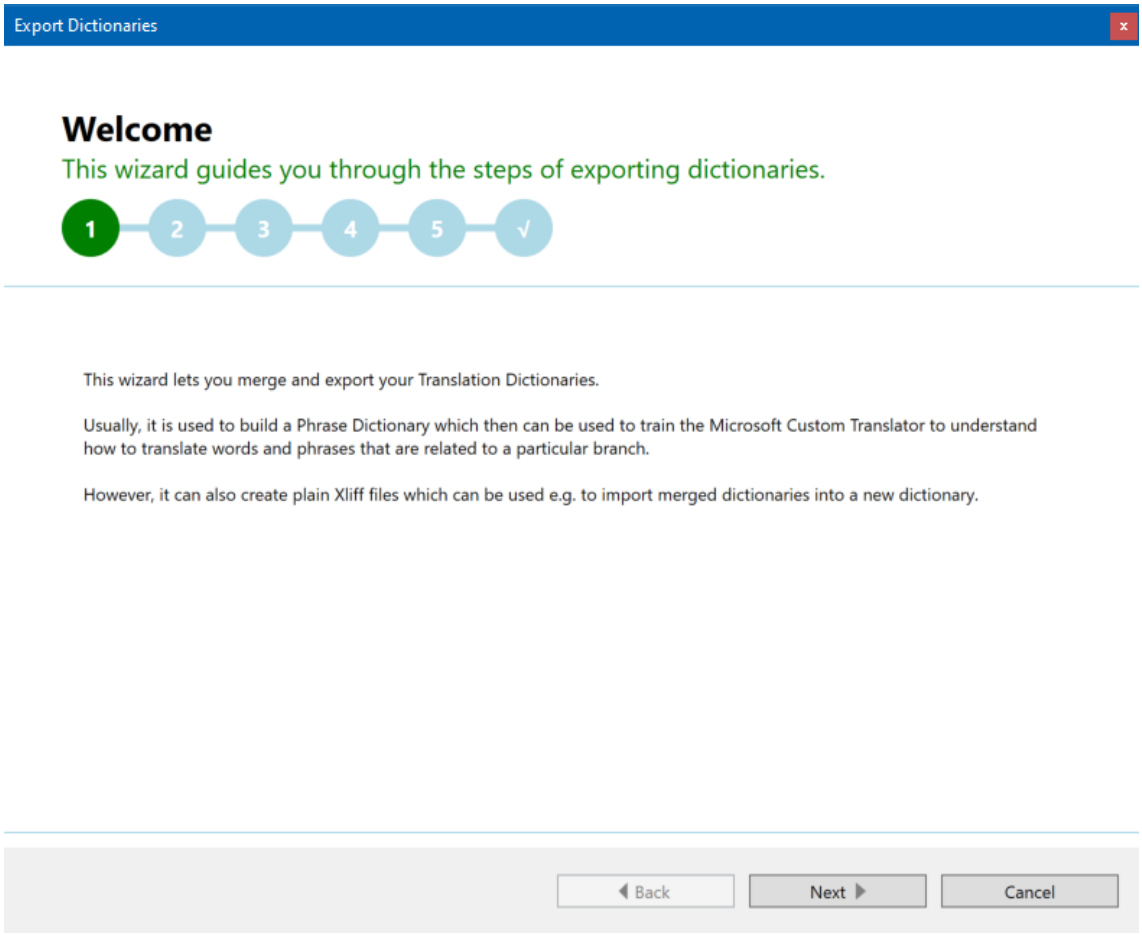
Then, start training a Microsoft Custom Translator model. Since a Custom Translator model can only trained once by using a phrase dictionary, the first step is to merge the appropriate dictionaries to one. Note, that this does not mean to merge the dictionaries into each other. Instead, you create an export file that will contain merged dictionary data. Then, you can use the exported file to create a Custom Translator model.

In Microsoft Custom Translator, creating a new project exposes a new category id. Within one project, there may be multiple models. However, there can be only one model active at a time and only one phrase dictionary can be used to create a model. Hence, each time you refine your branch dictionary, you need to repeat the steps explained above, i.e. each phrase dictionary upload needs to be a merged export of all required dictionaries. Since it is not possible to upload recent changes only, you must always include the entire phrase dictionary and create a new model from it.

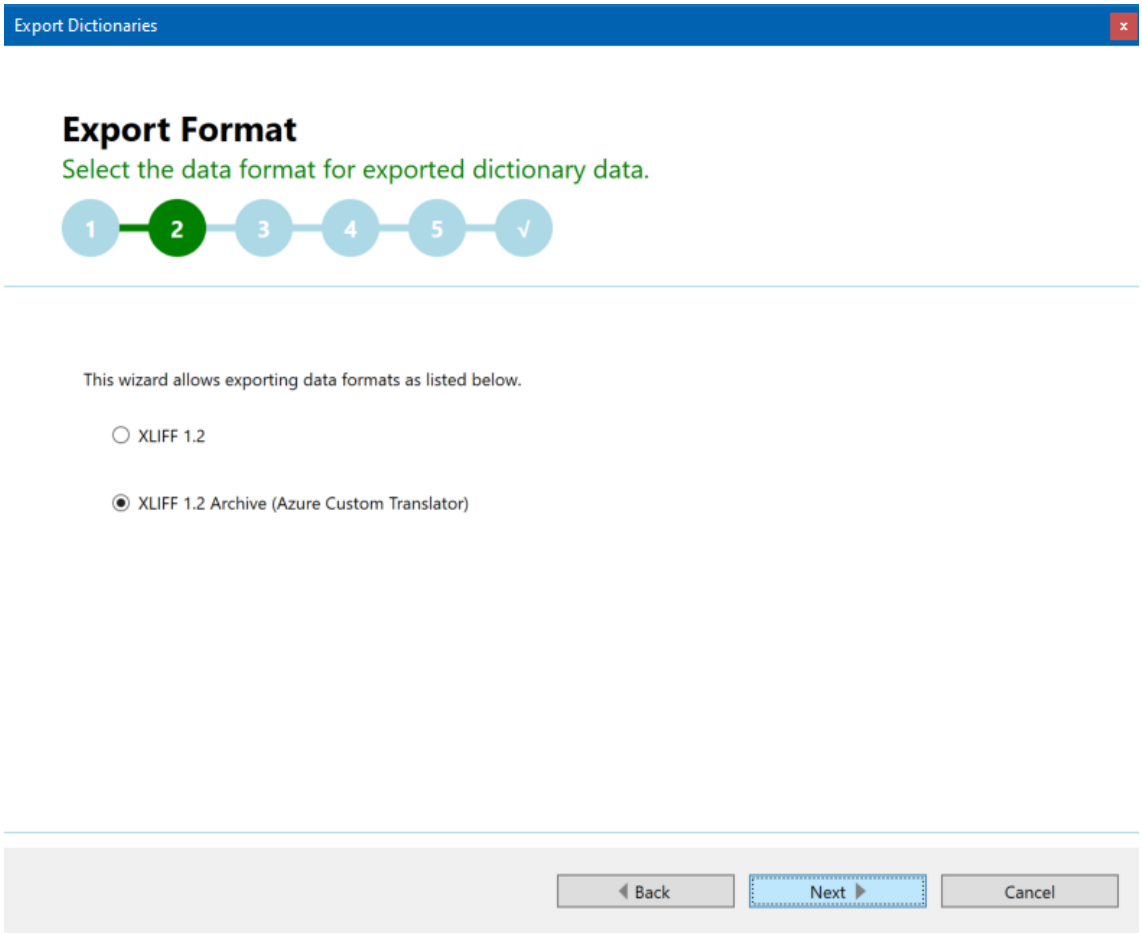
Note, that creating a new model is charged by Microsoft. You can examine the training costs in Microsoft Custom Translator before as well as after the model is created.

7.2 CREATING A MERGED DICTIONARY EXPORT FOR CUSTOM TRANSLATOR TRAINING

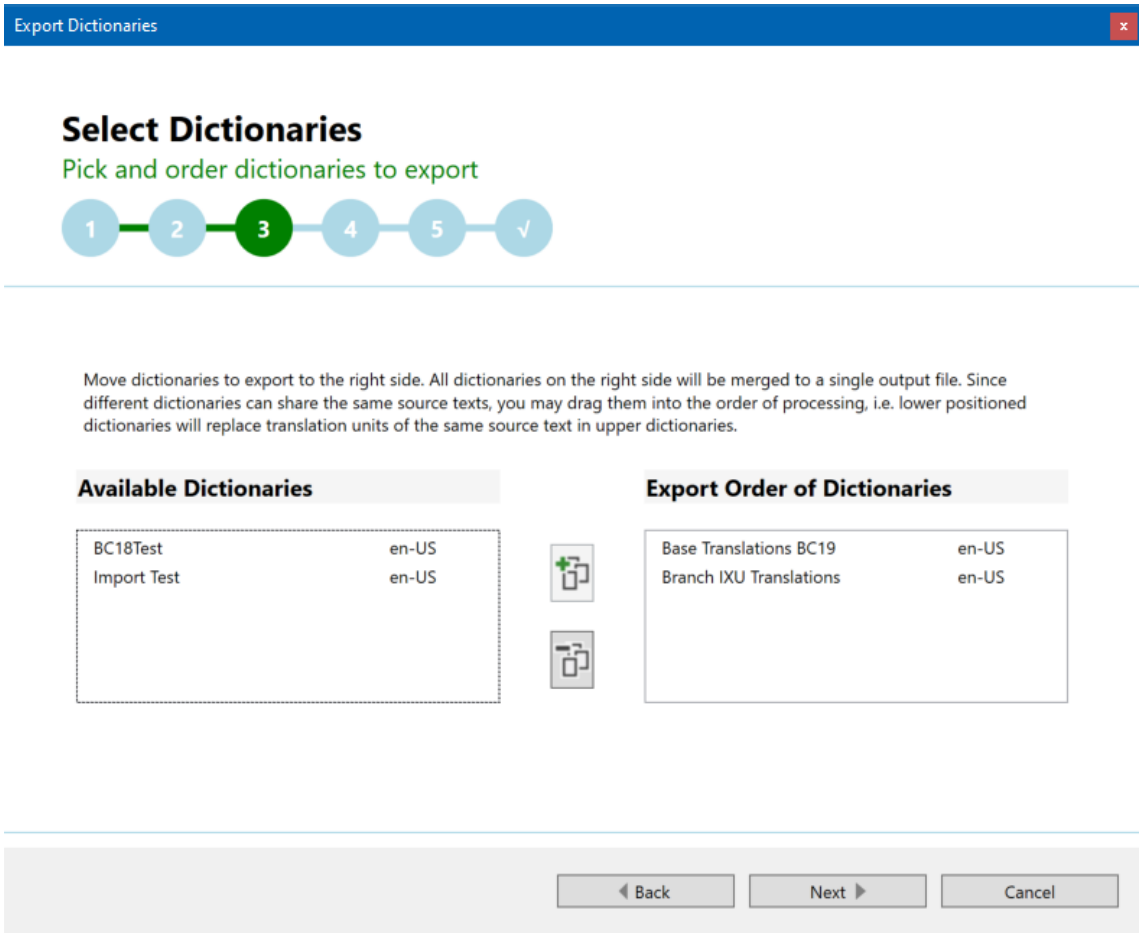
To create a merged dictionary export, click "Export" from the "Cloud Translation Dictionary" ribbon group. A wizard opens that guides you through all steps required for exporting dictionary data.



Click “Next”.



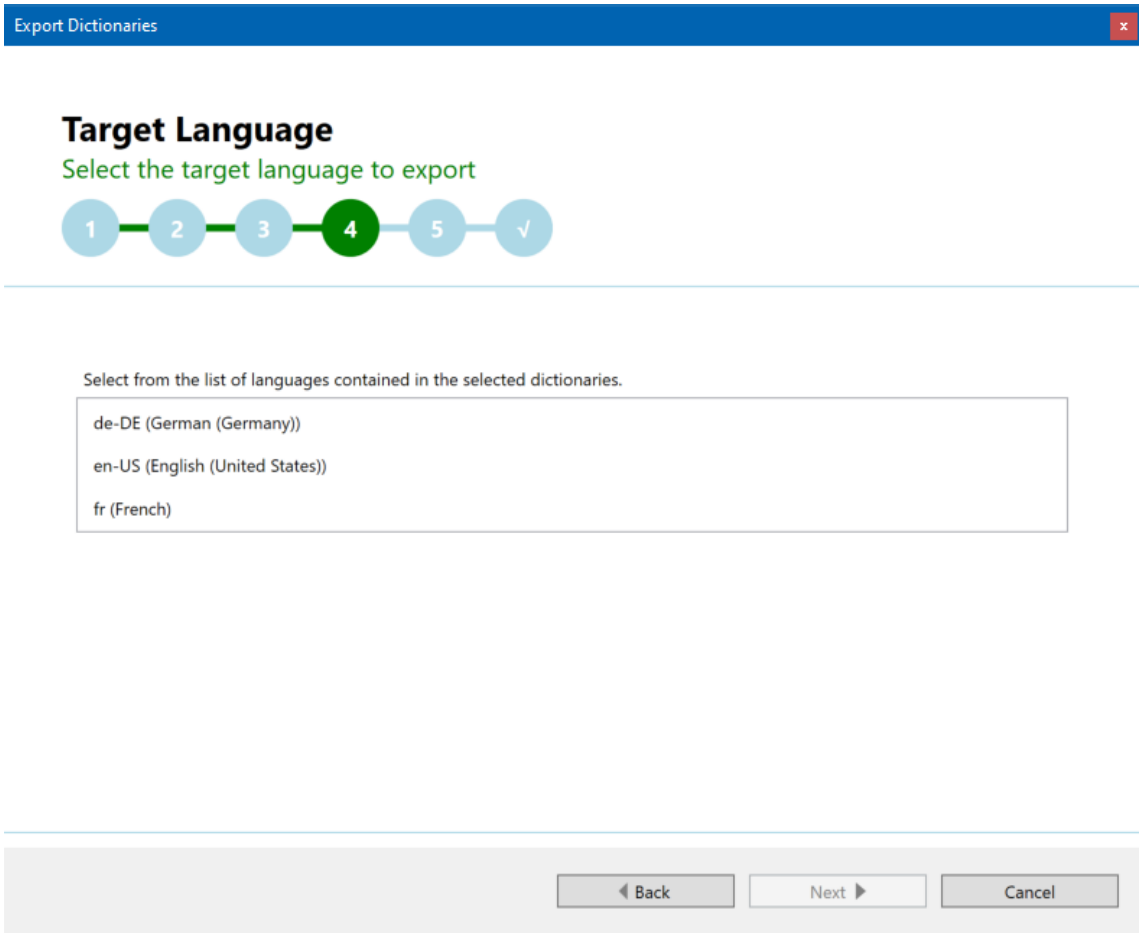
Select the option "XLIFF 1.2 Archive (Microsoft Azure Custom Translator)" to generate a file that is recognized by Microsoft Custom Translator as a phrase dictionary archive file. Then, click "Next".



In this wizard step, select the dictionaries to merge and export. By double clicking an available dictionary on the left side, each dictionary can be moved to the list of dictionaries to export in the right side list. Note, that you can also hold down the <Ctrl> key while you click all dictionaries to export with a single click one-by-one. Then, press the "Add" (+) button to move the selected dictionaries to the right in the same order as you selected them before.

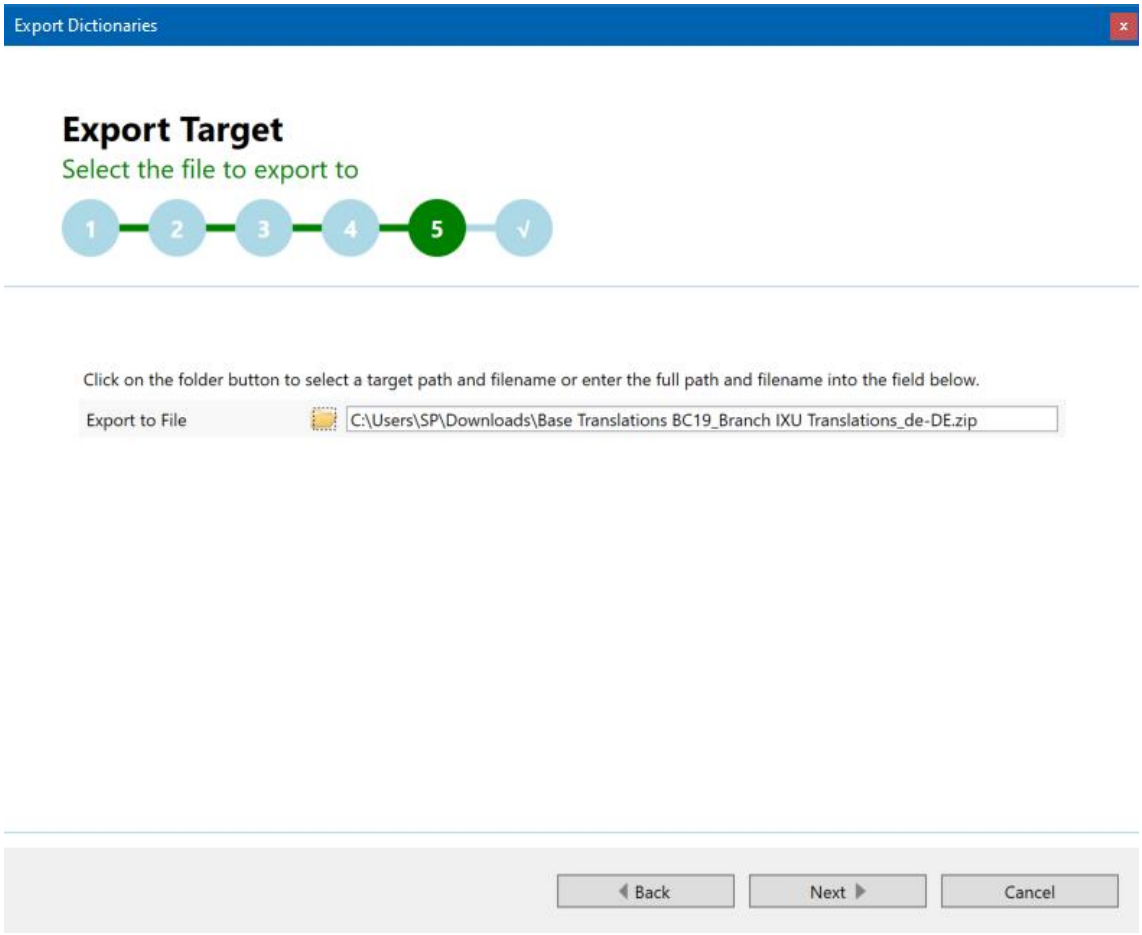
After adding all dictionaries to export to the right-hand list, you can drag & drop the dictionaries to the order in which you want them to be merged. Since different dictionaries can share the same source texts, each same source text entry from a lower positioned dictionary will replace the corresponding source text translation of all higher positioned dictionaries. Note, that existing translation units are replaced only if the replacing translation unit contains a translation.

Then, click "Next".



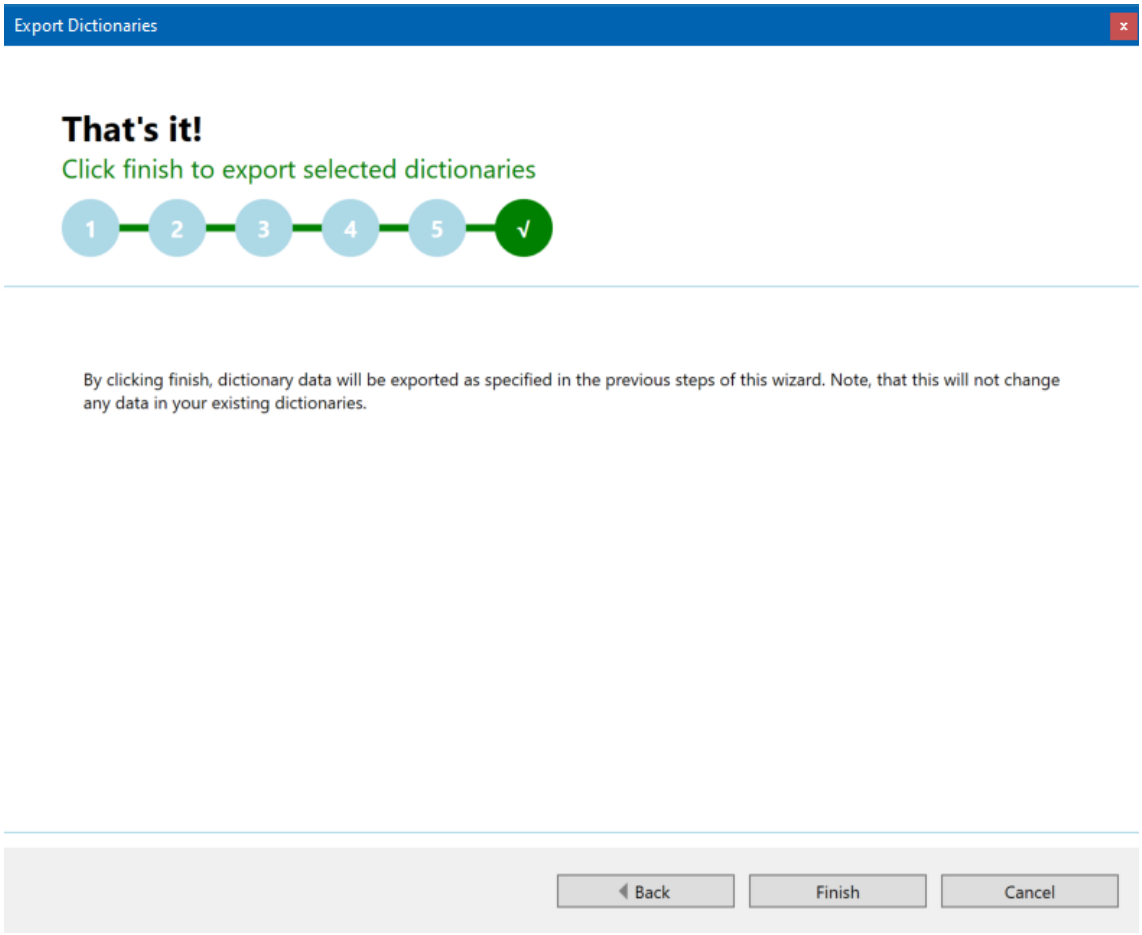
In this wizard step you select the single language you want to export. The list contains all languages that are contained in at least one of the dictionaries.

Select the language to export and click “Next”.



In this wizard step, select the file to export to. Click on the folder button to browse a folder and enter a file name. Alternatively, you may also paste the file path and name to the field.

Then, click “Next”.



In the last wizard step, click "Finish" to start exporting dictionary data to the file you specified above and wait until the wizard completes.

Locate the file created by the wizard and upload it to Microsoft Custom Translator.

7.3 Handling Custom Translations for Multiple Branches

In case you need to have various Translation Dictionaries e.g. when developing solutions for various branches, it is recommended to create one dictionary for each branch and use the same base app dictionary for all branches. You start creating the branch dictionary and export it as a merged dictionary as described above.

However, instead of uploading the merged dictionary file to the same project in Microsoft Custom Translator, you create a new project, one for each branch. Each project has exposes its own category id. Instead of copy and pasting this category id to the Translation Service Setup window in Qucamba Reports, you paste into to your current profile's Translation Category Id field. Whenever this setting is filled in, it will override the category from the general Translation Service Setup.

Then, upload the file to the new project, create a model from it and deploy the model.

8 CONCLUSION

Translating apps to foreign languages has become a much easier task than it has ever been before. The benefits of the demonstrated approach are based on a combination of

1. An easy-to-use translation text editor with clear-text display of translation unit paths
2. Starting with translations based on existing translations
3. Using a Microsoft Cognitive Services Text Translation resource
4. Training a translation model in the Microsoft Custom Translator
5. Preparing a clean and normalized subset of the basic terminology of the Business Central base app
6. Providing branch-specific phrases in a dictionary
7. Creating training material from a normalized translation dictionary
8. Advanced training of existing translation models

We demonstrated a practical way to achieve a fully automated app translation that requires a minimum effort on fixing unsuitable translations. Since the quality of this approach mostly depends on the quality and amount of training material, translation quality can even be enhanced later by adding more trainings and updating existing models.

Since each taken step in this approach effects all Qucamba Reports installations of the same license, there is no need of manually distributing any data or component to other installations.

9 TROUBLE SHOOTING

Problem	Solution
When using the VS Code extension NAB AL Tools simultaneously, all translation states get lost and tha NAB tool adds its own tags.	It's highly recommended to use only one single tool for synchronizing Xliff files. However, Qucamba Reports will remove the tags set by the NAB AL Tools extension automatically. To prevent the target states from being removed by the NAB AL Tools, add the following setting to each app's settings.json file: `"NAB.UseExternalTranslationTool":true`
Qucamba Reports doesn't show object names anymore. Instead, only hash values appear.	The app cache needs to be cleared. Go to the Qucamba Reports application menu and click "Manage App Cache…". From here, select "Clear Cache" or press <Ctrl> + <Shift> + <Del>. Then, restart Qucamba Reports and activate your profile again.
No translation units are displayed	Make sure you have no filters applied. Make sure you selected at least one language. Click the "Clear Filters" button in the "View" ribbon group or press Ctrl+Shift+F7 to remove all filters.

10 KEYBOARD SHORTCUT REFERENCE

Following table lists the most important shortcuts of the Qucamba Reports TRANSLATE module.

Shortcut Key	Description
Control + Shift + N	Create a new language file
Control + PageUp	Move to previous object
Control + PageDown	Move to next object
Control + Up	Move to previous translation in filter
Control + Down	Move to next translation in filter
Control + Y	Synchronize language files with the generated Xliff
Control + S	Save changes to underlying Xliff files
Control + Shift + F1	Auto translate entire app
Control + F1	Suggest translations
Control + F2	Set translation target state
Control + Shift + F7	Clear all filters and reset the current view
F5	Reload translations of the current app
F6	Set cursor into top Target Text field
F7	Set cursor into bottom Target Text field
F8	Set translation state to “translated”
F9	Set translation state to “to be reviewed”
F11	Open the Translation Id Hash Calculator
F12	Go to reference

11 RESOURCES

Microsoft Cognitive Services

- Portal: https://portal.customtranslator.azure.ai

Microsoft Custom Translator

- Introduction: https://www.microsoft.com/en-us/translator/business/customization
- Portal: https://portal.customtranslator.azure.ai
- Setup https://www.microsoft.com/en-us/translator/business/customization
- Setup Video: https://youtu.be/TykB6WDTkRc

Reference

- OASIS XLIFF Standard: https://docs.oasis-open.org/xliff/v1.2/os/xliff-core.html

Translation Pricing

- https://qucamba.com/de/purchase
- https://azure.microsoft.com/de-de/pricing/details/cognitive-services/translator