Localizing HTML Help

Philipp Strazny

Help systems are a good way of providing users of software, websites or manufactured products with structured additional information. HTML help (chm) or webhelp are the most commonly chosen formats because the content is authored in HTML. HTML, an open format and for many authors a familiar one, can be worked on with familiar text or HTML editors. Furthermore, it is fairly easy to repurpose HTML source content and, for example, to create PDF manuals from it.

In my six-year practice as a localization specialist for a translation agency, virtually all HTML help systems I received for localization were authored in RoboHelp. This article aims to provide practical guidelines for localizers, so I will concentrate mainly on localizing RoboHelp projects. Many tricks and techniques mentioned here should be transferable to other authoring tools, but some are quite specific to RoboHelp.

In spite of this focus on RoboHelp, this article should not be understood as an endorsement of RoboHelp as an authoring tool. The sole reason for this focus is the *de facto* status of RoboHelp as the big kid on the block among help authoring systems. According to my crystal ball, however, this is likely to change. Unless someone is going to put some serious development effort into RoboHelp — current version: X5 — it will probably be replaced by MadCap Software's Flare. As far as I know, the original creators of RoboHelp split off from eHelp/Macromedia/Adobe to start MadCap Software. Due to this fact as well as the apparent lack of further development at RoboHelp's current home, I am inclined to accept Flare as basically the next-generation RoboHelp.

For anyone with business interests outside of the United States, RoboHelp presents serious limitations. It relies on ASCII resource files, which preclude any localization into non-ASCII languages, and it puts localizable text items in many different and hard-to-reach places. These limitations are somewhat surprising in today's globally-minded world, especially since earlier versions of RoboHelp, namely X3, were more promising.

Nevertheless, even if the up-and-coming Flare is going to take a large market share away from RoboHelp, localizers will for years to come continue to be presented with the task of translating RoboHelp projects. In this article, I will try to summarize my experience with such tasks. This experience was often painful because some translatable text items are so well hidden that you may not be aware of them until your English RoboHelp project has been fully translated into Chinese and you suddenly stumble over English text strings in your supposedly final proofing.

The recipe for successful localization

Ingredients

RoboHelp. This is obvious to experienced localizers, but it is worth stating for newcomers. You cannot localize a RoboHelp project without having RoboHelp, ideally the same version that your clients have. Without RoboHelp you will miss some translatable text, and you won't be able to compile the foreign language Help, an important quality assurance (QA) step.

A translation tool that can handle tagged text. Again, this is obvious to experienced folks, but it must be emphasized for newcomers. Make

Philipp Strazny is a localization specialist at The Geo Group and editor of The Encyclopedia of Linguistics *(2004).*

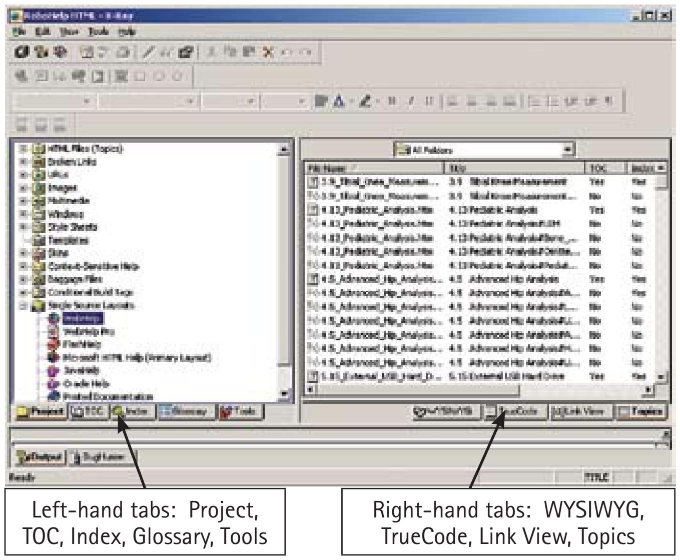

Left-hand tabs: Project, TOC, Index, Glossary, Tools
Right-hand tabs: WYSIWYG, TrueCode, Link View, Topics

Figure 1.

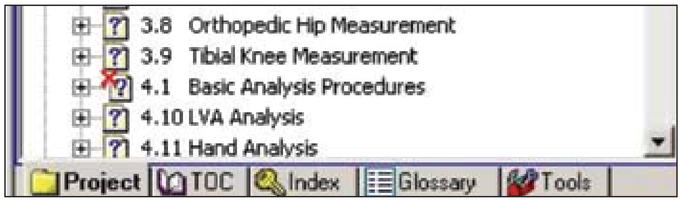
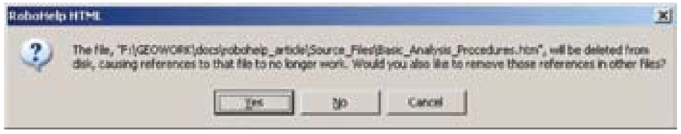

3.8 Orthopedic Hip Measurement
3.9 Tibial Knee Measurement
4.1 Basic Analysis Procedures
4.10 LVA Analysis
4.11 Hand Analysis
Project
TOC
Index
Glossary
Tools

Figure 2.


RoboHelp HTML

The file, "F:\GEOWORK\docs\robohelp_article\Source_Files\Basic_Analysis_Procedures.htm", will be deleted from disk, causing references to that file to no longer work. Would you also like to remove these references in other files?

Figure 3.

sure your translators do not attempt to get by with just Microsoft Word. I think that most current translation tools will do, as long as they list HTML as an acceptable input format and allow you to perform some customization. Since I do not want to endorse a particular tool, I will state what needs to be done. How exactly this is done depends on the specific tool being used.

A suitable image editor, usually of the Photoshop/Gimp variety.

The complete source directory as well as editable source files for all localizable images.

Basic Steps

Step 1. Source quality check

Step 2. Make copies of source directory

Step 3. Translate images

Step 4. Translate miscellaneous text items: browse sequences, chapter headings for printed documentation, window title

Step 5. Translate miscellaneous text files: table of contents, index, glossary

Step 6. Translate HTML source text

Step 7. Compile foreign language versions

Detailed steps in localizing RoboHelp projects

Step 1. Source quality check

RoboHelp translation projects are just like any other translation project, and the following formula holds:

garbage => garbage X number of languages

This means that you had better find any errors in the source files or you'll end up fixing them multiple times. It is much preferable to do a thorough check first. Your clients will be grateful if you can help pinpoint and fix problems in their help files, too.

After you receive files from your client, you can quickly check whether you have received the correct set of files. Among a number of other files, a RoboHelp source directory will contain:

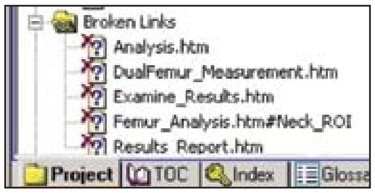


Figure 4.

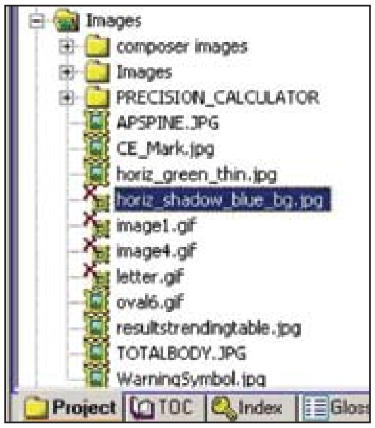


Figure 5.

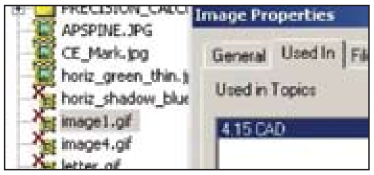


Figure 6.

- .htm files: the files holding the help content
- .jpg, .gif and other image files, potentially in relevant subfolders: the images displayed in the help pages
- .hhk: the index of the help system
- .hhc: the table of contents of the help system
- .glo: the glossary of the help system (optional)
- .xpj: the project file, that is, the file that tells RoboHelp how all these pieces fit together.

If the .xpj, .hhc and .hhk files are missing, your client likely sent you webhelp output files. You need to ask your client to zip the complete source directory, so you have everything you need to compile.

Assuming that the source file directory appears complete, you now need to open the project in RoboHelp. Double-click on the .xpj file, which opens the main RoboHelp window (Figure 1).

By default, the Project tab is displayed in the left-hand pane. Here you can verify whether RoboHelp has found all required resources. Check the folders in the Project tab for any indications of missing files.

HTML Files (Topics): expand this folder by clicking on the plus symbol. The subfolders mirror the actual folder structure in your source directory. If a file is missing, RoboHelp will display it with a red X (Figure 2).

Here, the file "Basic Analysis Procedures" is missing. If you right-click on this file name and select Delete, you might see a warning like the one in Figure 3.

This tells you that other files within the project contain references (hyperlinks) to the file Basic_Analysis_Procedures.htm, so it is possible that it was deleted from the file system inadvertently.

Broken links: If a file is missing that is referenced from other files, RoboHelp will also report broken links (Figure 4).

Images: If images are missing, they also appear crossed out. (Figure 5). You can double-click on the image file name to see where in the project this image is being used.

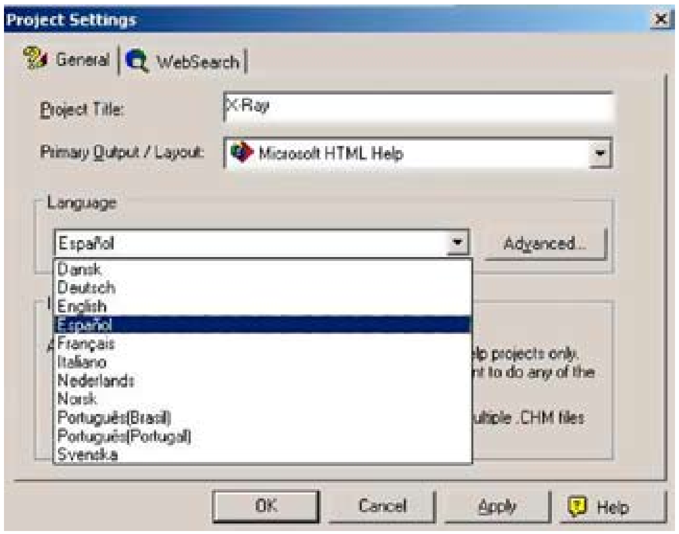
In Figure 6, you can see that image1.gif is used in topic 4.15 CAD.

Similarly, you should check for missing Multimedia files, Stylesheets, Skins and Baggage Files.

If you find evidence of missing files, the Help author would certainly like to know about this. Write up a short report and ask your client to fix these issues and to resubmit a new source package. If your client tells you to go ahead with the translation anyway, make it clear that the same problems will carry over into the translated versions, and any fixes later on will be more time consuming and costly.

Step 2. Make copies

Make a copy of the source directory for each target language. You will replace all English source files with translated versions. You also need to make an adjustment to the project settings, namely setting the target language.


Project Settings

General | WebSearch

Project Title: X-Ray

Primary Output / Layout: Microsoft HTML Help

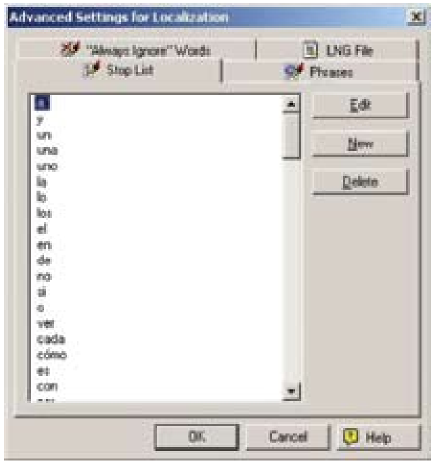
Language: Español

Dansk
Deutsch
English
Español
Français
Italiano
Nederlands
Norsk
Português(Brasil)
Português(Portugal)
Svenska

Figure 7.

The language drop-down list (Figure 7) tells you which languages RoboHelp supports. You'll see that the list is fairly limited. There are several reasons for this. It is a simple programming shortcoming that RoboHelp stores certain text items in ASCII files, so double-byte character sets are completely ruled out.

On the other hand, RoboHelp offers several nifty aids to the help author that rely on language-specific predefined lists of words. You can see these when you click on the Advanced button in the Project Settings dialog (Figure 8). As a localizer, you typically do not need to touch these.


Advanced Settings for Localization

"Always Ignore" Words | LNG File | Stop List | Phrases

a
y
un
una
uno
la
lo
los
el
en
de
no
si
o
ver
cada
cómo
es
con

Figure 8.

The LNG file contains user interface commands used by RoboHelp. It is possible to customize this file, but I would recommend not doing so. As stated in RoboHelp's online help, "if RoboHHRE.LNG is already in your end user's Windows directory, it will override the file in your Baggage Files folder." Thus, your customization efforts may be thwarted by interfering files on the end user's machine.

The other three word lists are used for search and indexing functions.

Step 3. Translate images

Images are handled just as in desktop publishing or website translation projects.

Use the thumbnail view in your Windows explorer to locate all images containing text in the RoboHelp source directory.

Help systems are usually about software, so there are likely to be screenshots. Try to convince your client to supply you with localized screenshots before you start translating. Having localized screenshots is the best way to ensure that the terminology of the help accurately matches that of the software.

At NCS a smile translates into service. Let us show you how.

Sara Jones - Project Manager
NCS Enterprises, L.L.C.

What communication means to NCS:

- **listening** to our clients' needs
- **asking** questions to make sure we get it right
- **thinking** ahead to avoid problems
- **providing** solutions

NCS

Translation that goes beyond words.

www.ncs-pubs.com Tel.: 412 278 4590

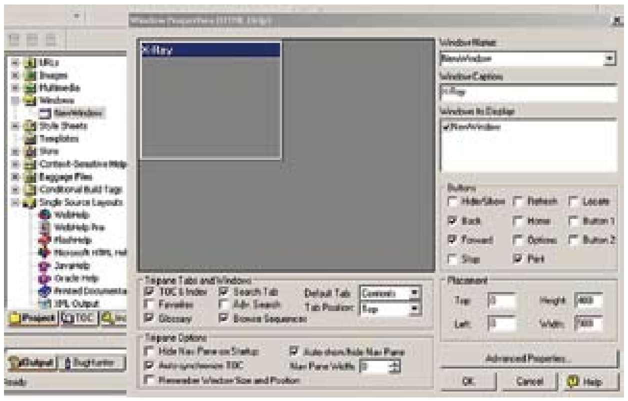
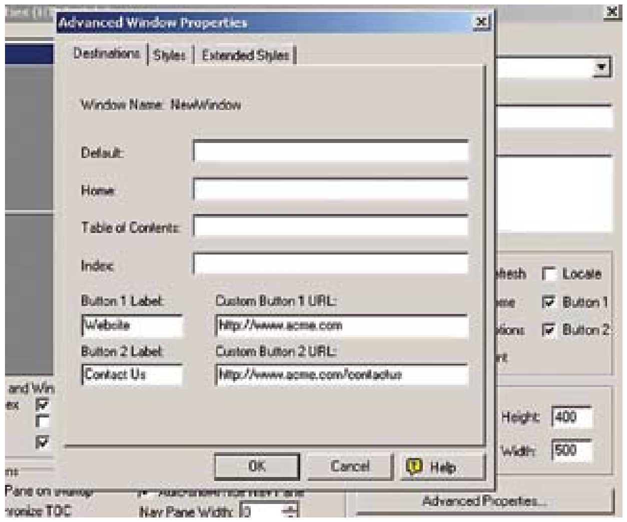


Figure 9.



Advanced Window Properties

Window Name: NewWindow

Button 1 Label: Website
Custom Button 1 URL: http://www.acme.com

Button 2 Label: Contact Us
Custom Button 2 URL: http://www.acme.com/contactus

Figure 10.

For other types of images such as buttons, diagrams and so on, try to get your hands on editable source files for Photoshop, Illustrator or similar programs. Otherwise, you'll have to lay the translated text over the source text and/or recreate the whole image — a time-consuming and thus costly alternative.

Extract all translatable text from the images. Unfortunately, this usually means manually copying and pasting from the image editor into an RTF.

Translate the RTF.

Reimport (cut and paste) the text back into the image source file.

Save the image with the same width and height dimensions and filename.

Copy the translated image over the source image in the RoboHelp source directory.

Step 4. Translate miscellaneous text items

There are a few text items that you will have to manually extract from the source project, and after translation, you'll copy and paste them into the appropriate locations of the relevant project file for the given language.

Window title. In the Project tab in the left-hand pane (Figure 9), expand the Windows folder and double-click on all window names listed there. This opens the Window Properties dialog for the respective window.

Copy and paste the Window Caption text into a text file for translation. If "Button 1" or "Button 2" is checked, click on Advanced Properties.

Copy the button labels into a text file for translation (Figure 10). These buttons are customizable additions to the standard help interface.

Chapter headings for printed documentation. It is possible that your client asks you to generate RTF or PDF printed documentation alongside the webhelp or HTML help output. If this is the case, double-click on Printed Documentation (Figure 11) under Single Source Layouts in the Project tab.

Your client should already have set up the parameters for the generation of printed documentation. Ideally, you would also have received a PDF of the original printed documentation generated by your client, so you know exactly what the expectations are. Generally, you can just accept the settings and click Next until you reach the Printed Documentation Content page. Here, you can specify

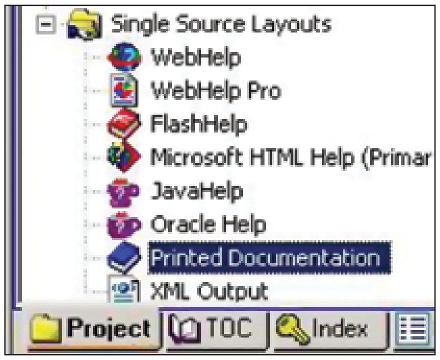


Figure 11.

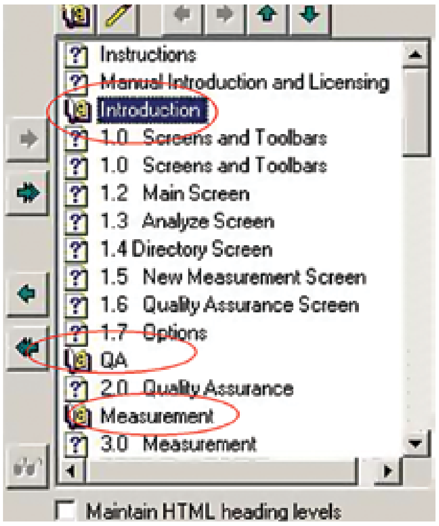


Figure 12.

which topics should be included in the printed documentation. Again, this should be configured already.

The table of contents (TOC) of the printed documentation generally follows the TOC of the HTML help, but watch out for additional chapter headings that can be inserted here. In this case, Introduction, QA and Measurement were added (Figure 12). These text items need to be copied and pasted into a separate text file for translation.

Browse sequence headings. "Browse sequences" are global navigational elements in compiled help. They are displayed at the top of the page and show the user where the current page fits into the overall layout of the help (Figure 13).

The individual page titles of the browse sequences are pulled from the TOC, but as for printed documentation, it is possible to add additional "chapter" headings. Otherwise, all pages would appear in one linear string.

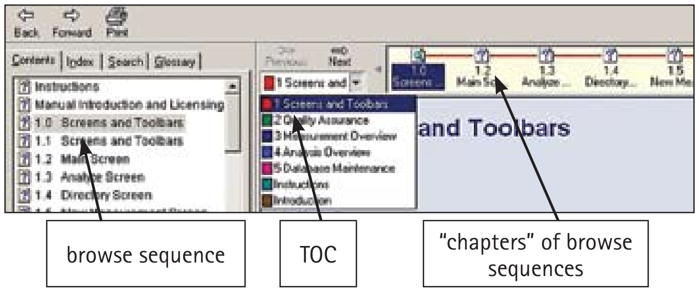


Figure 13.

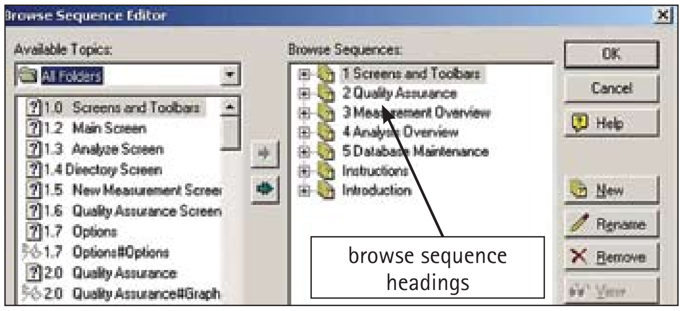


Figure 14.

These browse sequence headings are edited via Tools > Browse Sequence Editor (Figure 14).

Browse sequences headings are only editable through the Browse Sequence Editor interface, that is, for translation they need to be copied and pasted into a text file, and the translation needs to be copied and pasted back into the Browse Sequence Editor editor.

Note that RoboHelp displays browse sequence headings in alphabetical order. Your client is either happy with the alphabetical ordering of the source text or may have chosen headings that sort appropriately. Of course, the translated headings will likely appear in a different order, so you may have to prepend numbers to ensure the ordering preferred by the client. You may want to talk to your client about this in advance.

In the example here, the author of the English source text already ran into the same problem and solved it the same way, appending the numbers 1 through 5 to headings that would otherwise appear in a different order. Notice, however that “Instructions” and “Introduction” do not have numbers and may switch places after translation. This may or may not be a problem for the client, so you may have to discuss a strategy to guarantee a certain ordering.

Step 5. Translate miscellaneous text files

RoboHelp stores certain textual information in special files. Due to their idiosyncratic format, you may have to prepare them individually for translation. Your translation tool may have direct support for such files, in which case you can skip this section. If you do need to prep these files for translation, the required steps depend on your choice of translation tool. I will, however, introduce the formats and discuss what needs to be done.

There are three files, all of which are located in the root folder of your RoboHelp directory. These can be localized directly, that is, without having to go through the RoboHelp interface. Remember that these files are ASCII files. Unicode characters won’t work.

.hhc – the table of contents. The .hhc file is basically an HTML file, that is, it uses HTML tags to structure its content, as in the following code.

```
<html>
<!-- Sitemap 1.0 -->
<object type="text/site properties">
  <param name="SiteType" value="toc">
  <param name="Image Width" value="16">
  <param name="Window Styles" value="0x800002">
  <param name="ExWindow Styles" value="0x100">
</object>
<ul>
  <li><object type="text/sitemap">
    <param name="Name" value="What's New">
    <param name="Local" value="what_s_new.htm">
  </object>
  <li><object type="text/sitemap">
    <param name="Name" value="How to use your online help">
    <param name="Local" value="how_to_use_your_online_help.htm">
  </object>
  <li><object type="text/sitemap">
    <param name="Name" value="Title and CE">
    <param name="Local" value="title_and_ce.htm">
  </object>
  <li><object type="text/sitemap">
    <param name="Name" value="About This Manual">
  </object>
```

As with other tagged formats, you need to ensure that the overall format remains unchanged during translation, that is, the tags need to be protected. The only strings that need to be translated here are the value strings of param tags where the name attribute is set to “Name.” If your translation tool supports

FOLLOW UP
Your premier partner in Brazil!
Leadership is a consequence of sustained innovation!
We're ready to meet your localization goals in Brazil, with great ROI!
Tel.: +5521 2524-2994 • Rio de Janeiro • Brazil
www.follow-up.com.br

conditional tags, you may be able to set up a filter that gives you direct access to these strings.

If your translation tool doesn't support conditional tags, such as older versions of TagEditor, you may have to change the tag structure.

Replace all param name="Local with paramname="Local and change all param name in the top text/site properties element to paramname.

Translate only the value attribute of param tags.

Change all paramname back to param name.

Copy the translated .hhc file over the original file and then open your RoboHelp project. Open the TOC tab in the left-hand pane and verify that the table of contents appears correctly.

.hhk – the index. The .hhk file has the same structure as the .hhc file, so the translation process is also the same.

```
NAME=2D
2 dimensional
NAME=3D
Three-dimensional
NAME=A/P
Anterior/Posterior anatomical coordinates
NAME=ACGD
Advanced Control Gradient Driver
```

After translation, copy the translated .hhk file over the original file and then open the RoboHelp project. Open the Index tab in the left-hand pane and verify that the index appears correctly. If the project is set up with a binary index, RoboHelp automatically alphabetizes the index. Otherwise, right-click into the Index tab page and select Sort > Current level and below from the popup menu.

.glo - the glossary. The .glo is a simple text file.

During translation, you need to watch out that you keep the line breaks and don't insert any additional line breaks. Each glossary item occurs on two lines. The first line shows the glossary entry, introduced by NAME=, and the following line provides the explanation. You could set up a filter in your translation tool, or, if you work with a TRADOS-compatible Word plug-in, you can use this rather low-tech approach for translation:

Open the file in Word.

Do a search & replace for "NAME=" and replace it with "NAME=" with the style set to tw4winExternal. Your plug-in should then ignore the "NAME=" part altogether.

After translation, save the file as a plain text file with *.glo extension.

Copy the translated .glo file over the original file and then open your RoboHelp project. Open the Glossary tab in the left-hand pane and verify that the glossary appears correctly. RoboHelp will alphabetize it automatically.

Step 6. Translate HTML source text

Now you should enter familiar territory. Translating the *.htm source files of a RoboHelp project is just like translating a simple static website. All links within the project are relative, which means that you can open any of the .htm files in a browser and the links should work as long as you do not change the directory structure. Also, the root directory of the RoboHelp source package contains stylesheets (*.css) and javascript (*.js) files that are important for display and functionality. Last, all images are in their appropriate locations.

When you send files to your translators, you should make sure that they receive all HTML files in their intact folder structure as well as stylesheets, javascript and images. This will provide them with the appropriate context when they view the individual pages in their browser for reference. This means that you should make a copy of the full source directory and then delete all files that are of a different filetype – everything but *.htm., *.css, *.js, *.gif/jpg/bmp/png and so on.

Unfortunately, your clients, like most humans, are lousy housekeepers. When you asked them to just zip the RoboHelp source directory and send it to you, they probably did just that, that is, the source directory that you have in front of you may contain files that are not even being used. This can occur when files were externally created and never added to the current project. You can use this simple trick to remove all unused .htm:

Open the Topics tab in the right-hand pane.

Select all topics.

Right-click into the Topics tab and select Properties from the context menu.

Open the Status tab in the Topic Properties window, select a Status from the Status drop-down list, click Apply. Then select a different Status and click Apply again. This should ensure that every single topic file has been modified. Note the current time.

Open your Windows explorer and browse to your source directory.

Search for all .htm files that were modified before the time you noted above. Delete these files and translate only the remaining ones.

Remember that I recommended that you send the .htm files to your translator in their intact folder structure. This is for the translator's benefit because all links will work, the styles will be applied as they should, and the images will be displayed. Yet you should insist that the translator send the translated files back to you in their intact folder structure as well. Also – very important – the translator should not change the file names. After going through your usual QA steps, you can then copy the directory with the translated files and copy it over your source directory. This way you just reopen the RoboHelp project and all your translated files are in place.

Step 7. Compile foreign language versions

If you've carefully performed all previous steps, compiling is a routine step. Simply click on File > Generate Primary Layout and click Finish in the appearing dialog, thereby accepting all settings.

A note on supported languages:

I mentioned file format limitations as well as language-specific features in RoboHelp. These are all good reasons to localize RoboHelp projects only into languages that are explicitly supported by RoboHelp.

You can go into unsupported languages, but would have to do so without RoboHelp. Here I will mention just these possibilities:

You could port an X5 project down to X3 to be able to use, for example, the Asian versions of RoboHelp that were available for X3.

You could port an HTML help project to Microsoft's HTML Help Workshop, but you would lose all of RoboHelp's fancy features, such as index search, global search, browse sequences, conditional text and so on. And localizing HTML Help Workshop projects is quite painful because for many (double-byte) languages, the system default language needs to be changed for compilation, which means constant rebooting.

You could port the RoboHelp project to MadCap Software's Flare, which claims to be able to seamlessly import them, but I have not had the chance to try it yet. **M**