LISTSERV Maestro Admin Tech Doc 17

UI Customization and Translation

May 26, 2014 |
lsoft.com

This document is a LISTSERV Maestro Admin Tech Doc. Each admin tech doc documents a certain facet of the LISTERV Maestro administration on a technical level. This document is number 17 of the collection of admin tech docs and explains the topic “UI Customization and Translation”.

Last updated for LISTSERV Maestro 6.0-2 on May 26, 2014. The information in this document also applies to later LISTSERV Maestro versions, unless a newer version of the document supersedes it.

All of L-Soft's manuals for LISTSERV are available in ASCII-text format via LISTSERV and in popular word-processing formats via ftp.lsoft.com. They are also available on the World Wide Web at the following URL:

URL: http://www.lsoft.com/manuals.html

L-Soft invites comment on its manuals. Please feel free to send your comments by e-mail to: MANUALS@LSOFT.COM

Table of Contents

1 Simple Text Customization 1

2 User Interface Translation 2

2.1 Requirements for a Translation 2

2.2 Installing the Resource Translation Tool 2

2.2.1 Starting the Resource Translation Tool on Windows 3

2.2.2 Starting the Resource Translation Tool on Linux or Solaris 3

2.3 Translation Kits 3

2.4 Translating 3

2.4.1 Importing Translations from Previous Versions of LISTERV Maestro 4

2.4.2 Navigating in the Resources Tree 4

2.4.2.1 Quick Navigation With Keyboard Shortcuts 4

2.4.2.2 Finding a Node in The Resources Tree 5

2.4.2.3 Keyboard Shortcut Summary 5

2.5 Installing a Translated Language 5

2.5.1 Installing a Translation for the Administration Hub or the LUI Main Application 5

2.5.2 Installing a Translation for the Maestro User Interface Subscriber Pages 6

2.6 Using a Translated Language 6

2.6.1 Languages for the Administration Hub and the Maestro User Interface Main Application 6

2.6.2 Languages for the Maestro User Interface Subscriber Pages 6

2.7 General Concepts 7

2.7.1 Key/Value Entries 7

2.7.2 HTML Code 8

2.7.2.1 HTML Escape Sequences 8

2.7.2.2 HTML Whitespace and Linebreaks 8

2.7.2.3 HTML Formatting in Text Resources 9

2.7.2.4 Non-HTML Exceptions 10

2.7.3 Word Replacement 10

2.7.3.1 Standard Numbered Placeholders 11

2.7.3.2 Named Placeholders on the LISTSERV Maestro User Interface Subscriber Pages 12

2.7.3.3 Placeholder Validation 12

2.7.4 Miscellaneous 13

2.7.4.1 Date and Time Formats 13

1 Simple Text Customization

Almost all texts in the LISTSERV Maestro User interface can be translated and customized with the Resource Translation Tool that is described in the next section. A small subset of text strings can also be customized in a simpler manner, as described in the following.

Create a text file called "`custom.properties`" which must be located in the folder

```
[maestro_install_folder]/lui
```

This file must be a text file that follows the rules of LISTSERV Maestro INI-files (see "LMA Admin Tech Doc 1 – Configuration Files"). This means, that the file consists of entries of the form "`key=value`", with which you can define a limited number of customized texts.

Each custom text string consists of a "`key=value`" pair where the key is as listed below, and the value is the text that is supposed to appear in the user interface (and which has to follow the INI-file rules).

The following text string keys are currently available for customization:

`app.customizable.title.titleBar=YOUR_TEXT` will show "`YOUR_TEXT`" as the title text in the blue header bar that is shown at the top of each page, for a user in normal (non-evaluation) mode. The blue header bar will be the background behind the title text, which is drawn with a white, bold-faced, italic font.

If you do not include this key, the text "LISTSERV Maestro X.X" will be shown (where X.X is replaced with the current version number).

`app.customizable.title.titleBarEvaluation=YOUR_TEXT` has more or less the same effect as "`app.customizable.title.titleBar`" above, but this title text is shown for a user in evaluation mode (instead of the normal title text).

If you do not include this key, the text "LISTSERV Maestro X.X - Evaluation" will be shown (where X.X is replaced with the current version number).

`app.customizable.msg.headerText=YOUR_TEXT` will add "`YOUR_TEXT`" as a header text to the blue header bar that is shown at the top of each page, to the right of the standard title text in that bar. The blue header bar will be the background behind the text, which is drawn with the same font as the standard title text in the header bar.

If you do not include this key, no such text will be shown.

`app.customizable.msg.footerText=YOUR_TEXT` will add "`YOUR_TEXT`" as a footer text at the bottom of the page. The footer text will be drawn at the bottom of each page, with the standard small page font.

If you do not include this key, no such text will be shown.

`app.customizable.url.maestroProductPage=YOUR_URL` will define "`YOUR_URL`" as the URL that is accessed if the user clicks on the product logo icon in the blue title bar (at the top of each page) or on the version/build number that is shown in the grey bar on the login page.

If you do not include this key, the URL http://www.lsoft.com/products/maestro.asp is used.

2 User Interface Translation

This section gives an overview of how to translate the texts of the LISTSERV Maestro user interface to another language (or how to customize the English language texts), using the L-Soft Resource Translation Tool. It assumes a good acquaintance with computers in general and how to work with files in the file system.

The description in this section was last updated for version 1.0-4 of the Resource Translation Tool.

2.1 Requirements for a Translation

To be able to translate the LISTSERV Maestro user interface, the following basic requirements must be met:

- You need to have access to a server with an installation of LISTSERV Maestro.
- You need to have an installation of the L-Soft Resource Translation Tool.
- You need to have a translation kit that matches the LISTSERV Maestro version that you have installed.
- You need a good understanding of editing HTML code (and the meaning of the various HTML tags and how to escape reserved HTML characters).

2.2 Installing the Resource Translation Tool

The Resource Translation Tool is available for download in the form of a ZIP file. Unpack the file to a suitable location on your disk. In the following, this target folder is called "installation folder". When unpacking, make sure to preserve the folder names that are used in the ZIP file.

Note for Linux and Unix users: Also preserve file permissions when unpacking the downloaded file to make sure that the shell script mentioned below is executable. Otherwise, use `chmod` to modify the shell script file attributes manually to make it executable.

Unpacking the tool ZIP file creates several subfolders in the installation folder:

- `lib`: Contains the actual JAR file with the Java classes of the tool and additional JAR files from L-Soft and other sources.
- `Unix / Windows`: These subfolders contain additional files that are needed to start the translation tool on the supported operating systems.
- `work`: The translation kit(s) and translated JAR files will be placed here.

Note: To upgrade an existing an installation of a previous version of the Resource Translation Tool, unpack the downloaded file to the installation folder of the previous version, overwriting existing files. This way, previous translations that were created with the previous version are automatically available as import source when working on a new translation.

The resource translation tool is a graphical Java application and is deployed without an embedded Java Runtime Environment (JRE), to save download bandwidth. The tool was developed for Java 7 and checks upon startup if a compatible JRE is available on your computer. A compatible JRE can be downloaded from http://www.oracle.com. Alternatively, if you are starting the tool on a computer that has an existing installation of LISTSERV Maestro 5.0 or later, you can safely use the Java installation that comes with this version of LISTSERV Maestro. Do so by adjusting the startup command file to use the full path to the "`java`" subfolder of the installation folder as location for Java (see the comments in the command file for details).

2.2.1 Starting the Resource Translation Tool on Windows

Navigate to the "`Windows`" subfolder of the installation folder and double-click the command file "`TranslationTool.cmd`".

2.2.2 Starting the Resource Translation Tool on Linux or Solaris

Navigate to the "`Unix`" subfolder of the installation folder and execute the shell script "`TranslationTool.sh`". If problems with the Java installation are detected, the script reports these problems on the console. To be able to see the error message output, open a terminal window and execute the script from there. Once the script starts the tool without errors, it is safe to start the script without previously opening a terminal window.

2.3 Translation Kits

Before starting the tool for the first time, you need to acquire a translation kit file for the LISTSERV Maestro version that you want to translate (or customize), e.g. by downloading it from http://www.lsoft.com. Or contact support@lsoft.com.

The translation kit for LISTSERV Maestro comes in form of a ZIP file for each LISTSERV Maestro version.

For example, the file "`ListservMaestro6.0-1-TransKit.zip`" is for LISTSERV Maestro 6.0-1.

Once you have obtained the translation kit for the correct Maestro version, put the translation kit ZIP file into the "`work`" subfolder of your Resource Translation Tool installation folder.

The translation kit contains several components, one for the LISTSERV Maestro Administration Hub, one for the LISTSERV User Interface Main Application and finally one for the LISTSERV User Interface Subscriber Pages. Each of the available components supports a list of target languages.

2.4 Translating

Start the translation tool. Upon startup, the tool performs an integrity check of the translation kit files in the tool work folder and shows a drop-down list of the LISTSERV Maestro versions for which a valid translation kit has been found in the work folder. To begin with your translation, select the LISTSERV Maestro version, then select the translation kit component and finally the target language. Click OK to begin translating.

The main window of the translation tool is divided in two main areas, the left area contains the resource key tree of the selected translation kit component and the right area is divided in an upper and a lower part. The upper part displays the original text for the selected resource key, the lower part displays the currently defined custom text for the selected target language (initially empty).

Clicking the Save button prompts the tool to create an appropriate resources jar file which is named as follows:

`hubResources_[lang].jar`	(for the HUB)
`luiResources_[lang].jar`	(for the LUI main application)
`luiSubscriberResources_[lang].jar`	(for the LUI subscriber pages)

(where `[lang]` is replaced with the two-letter ISO code of the selected target language)

These files are written to the subfolder `translationsV[version]` of the translation tool work folder. For example, if you have chosen to translate the LUI subscriber pages of LISTSERV Maestro 6.0-1 to German, the folder is named `translationsV6.0-1` and contains (among others) the file `luiSubscriberResources_de.jar`.

Note: All components also list "English" as an available target language, which allows you to use the translation tool to customize user interface texts without actually performing a translation. All descriptions in this document are applicable without changes to the target language "English".

2.4.1 Importing Translations from Previous Versions of LISTERV Maestro

If the translation kit and the output folder for a previous version are present in the work folder and if the corresponding resources jar file for the selected target language is present for the previous version, too (see above), then the import button in the bottom right corner becomes enabled. Click the button to open a selection dialog that lists all available previous versions for the currently selected translation kit component. If, for example, currently the French translation of the subscriber pages of LISTSERV Maestro is selected and if the tool work folder contains a subfolder "`translationsV5.0-24`" which in turn contains a jar file named "`luiSubscriberResources_fr.jar`", then the version 5.0-24 is listed in the selection dialog.

Select the desired version and click OK. The import transfers the translations for all unchanged properties to the current version. "Unchanged" here means that the given property existed in the previous version and that its original text has not been changed by L-Soft during the transition from the imported version to the current version. The import uses this criterion to decide if the translation for a property that was present in the previous version can safely be imported to the current version. If on the other hand L-Soft *did* change the original text for a property or added a property, then an existing translation for the property is not simply imported to the current version. Instead, the import marks the property node in the resources tree with a yellow asterisk. When such a node is selected, the right pane shows four texts instead of the usual two: The upper two texts are the imported versions of the original text and (if present) the imported version of the translation. Next follow the usual two texts, i.e. the current version's original text and the editable text field for the current version's translation. Looking at the upper three texts should assist you in deciding if the imported translation can be used as the translation for the current version and if you now want to supply a translation for a key that did not require a translation previously.

The yellow asterisks remain in place during your current edit session. If you exit the tool and re-open it, the asterisks disappear. By once again importing the translations from the same version again, the yellow asterisks are once again created. This and all subsequent imports will not overwrite the changes that you have made in your current edit session.

2.4.2 Navigating in the Resources Tree

Translating or customizing texts for LISTSERV Maestro requires editing texts that are distributed over a considerable number of property keys in the translation kit, and frequently the texts are only short text snippets which can be translated quickly each on its own.

2.4.2.1 Quick Navigation With Keyboard Shortcuts

To select the next property key for translation, the translation tool supports standard mouse navigation. However, if many texts need to be edited, navigation via keyboard shortcuts can greatly reduce the time that is required for the translation as a whole.

Assume that you are working on the translation for one of the property keys, i.e. your edit focus is in the editor for the custom text of a property key. Use the standard keyboard keys to navigate in the text that is displayed in the editor. If you now want to finish editing the text for the current property key and want to edit the text for the next property key, press CTRL+ENTER or CTRL+DOWN. These

keyboard shortcuts locate the next property key node (i.e. the one immediately following the one that you were currently editing) from the resources tree and automatically select it for editing, i.e. the editor is refreshed with the current custom text for the new property key. Analogously, if you want to review the text for the next-upper property key, press CTRL-UP.

2.4.2.2 Finding a Node in The Resources Tree

Working on a translation frequently requires that you locate a particular properties key node in the resources tree. Finding the node can be accomplished by manually navigating in the resource tree by expanding and collapsing tree nodes and scrolling downwards until you have found the node that you are looking for. To speed up this task, the tool comes with a find dialog. Click the "Find" button in the bottom left corner of the main tool window to open the find dialog. A node can be found by supplying a part of the property key, by supplying a part of the original text or by supplying a part of the translated text.

2.4.2.3 Keyboard Shortcut Summary

Shortcut	Function
`ALT-F`	Open the "Find" dialog, or, if it is already open, begin searching.
`ALT-C`	Close the "Find" dialog
`ALT-N`	Find the next occurrence of the previous search string.
`ALT-S`	Save the Changes
`ALT-E`	Exit
`CTRL-ENTER` `CTRL-DOWN`	When editing a custom text, selects the subsequent properties key node for editing
`CTRL-UP`	When editing a custom text, selects the previous properties key node for editing

2.5 Installing a Translated Language

To install a translated language into an existing LISTSERV Maestro installation, simply copy the JAR file that you have created during translation to the "`lib`" folder inside of your LISTSERV Maestro installation folder. (It is recommended to keep the copy of the JAR file in your work folder even if you have completed the translation. This allows you to import your translation into the translation for a future version of LISTSERV Maestro, see above for details about the import capabilities of the translation tool.)

If you are installing the JAR file for the selected component and target language for the first time, you should not find a file with a conflicting name. If you are updating an existing translation JAR file, simply replace the existing file with your new version.

2.5.1 Installing a Translation for the Administration Hub or the LUI Main Application

If you have translated the Maestro User Interface Main Application component of the translation kit (which for example results in the tool writing the file `luiResources_de.jar` if your target language was "German"), then you must make this file available to the Maestro User Interface server **and** the Administration Hub server, i.e. copy the file into the "`lib`" folder(s) of the corresponding server(s) (this may in reality be only one server if you decided to install the two components on a single computer, so in such a case you would have to copy the JAR file only once).

If you have translated the Administration Hub component of the translation kit (for example yielding the file `hubResources_de.jar` if your target language was "German"), then you only have to copy the JAR file into the "`lib`" folder on the server where the Administration Hub is installed (i.e. do not copy the file to the Maestro User Interface server).

Restart LISTSERV Maestro on all servers where you put your new JAR file.

Note: If you are installing an updated version of the same resources file that is already in use by the running application, then you may observe that the operating system holds a lock on the file and disallows to replace it. In this case, you first have to stop the application, then replace the file, then restart the application.

2.5.2 Installing a Translation for the Maestro User Interface Subscriber Pages

If you have translated the Maestro User Interface Subscriber Pages component of the translation kit (for example the tool has written the file `luiSubscriberResources_de.jar` if your target language was "German"), then you have to copy the JAR file into the "`lib`" folder on the server where the Maestro User Interface is installed.

Note: In contrast to the behavior for the translations of the main application of the Maestro User Interface and the Administration Hub (see above), the JAR file for the subscriber pages **can** be replaced at runtime. It is **not required to restart** the application, however, to make it available to the system, log in to the Administration Hub and click on Global Component Settings → Maestro User Interface → General Administration. Click the **[Refresh Translations Now]** button on the page to prompt the system to freshly load the JAR file that you just copied to the lib folder.

2.6 Using a Translated Language

All languages that are installed are **cumulative**, e.g. they will all be available at the same time, in parallel.

This means, that on any LISTSERV Maestro installation, the default language "English" will always be available, and in addition, all other languages which were translated and installed as described above are also available.

2.6.1 Languages for the Administration Hub and the Maestro User Interface Main Application

LISTSERV Maestro will automatically determine the correct language for any user accessing the user interface with a web browser: The web browser sends the locale information of the user to LISTSERV Maestro, and LISTSERV Maestro presents the user interface using the language matching this locale, or, if no matching language is found, using the default language "English".

So to choose a language in Maestro, you need to tell your web browser the language that you want to use

2.6.2 Languages for the Maestro User Interface Subscriber Pages

In contrast to the rest of the user interface of LISTSERV Maestro, the subscriber pages of a given dataset ignore the locale information that is sent by the web browser but instead use a fixed language that is determined differently:

To define the default language for the subscriber pages, use the following setting in the file `lui.ini`:

```
DefaultCustomizationLanguage=XX
```

where you replace "*XX*" with the two letter lowercase code of the desired language. For example "`en`" for "English" or "`de`" for German.

If this key is not specified in the `lui.ini`, "English" will be used as the default language.

(You need to restart LISTSERV Maestro to make the changes effective.)

To individually define the language for the subscriber pages of a certain dataset, open the dataset settings wizard and change the language in the “Advanced Settings”, then save the changes.

2.7 General Concepts

The following subchapters describe some general concepts that need to be understood when performing a translation with the translation tool.

It is important that you read through all of them and that you understand and employ the concepts and rules explained.

2.7.1 Key/Value Entries

Even though the translation tool presents the translation kit contents in a tree-like structure, LISTSERV Maestro requires that the resource keys are available in the form of “key/value” pairs in a collection of text files that are bundled in a JAR file. The tool frees you from the tedious task of creating such a collection manually, but a general understanding of how resource keys and their values are used in LISTSERV Maestro is useful anyway.

Each resource key pair looks as follows:

```
<KEY>=<VALUE>
```

where “`<KEY>`” is replaced with the name of the key and “`<VALUE>`” with the language specific value associated with this key. Example:

```
app.caption.cancel=Cancel
```

This defines a key called “`app.caption.cancel`”, which has a value of “`Cancel`”.

And this is how it works: Whenever some language dependent text needs to appear in the user interface, the program simply contains the key for that text and a replacement directive. So when Maestro needs to display a page containing this text, it finds the directive: “Display the text associated with the key XYZ”. LISTSERV Maestro then looks up the key in the resource bundle and displays the value associated with this key.

So simply by changing the value of a key, you can change what LISTSERV Maestro will display. If abused, this means that you can make LISTSERV Maestro display any text you want, even non-meaningful or misleading text. However, this would surely not make much sense, since it would render the application quite unusable. Instead, you should wisely change the values so that their meaning is preserved.

If you decide to not supply the value for a key in your translation, LISTSERV Maestro will look up the original value. If you use the translation tool to supply custom **English** texts for resource keys (because you only want to customize a few selected texts in the UI), then it is sufficient to only supply the values for the keys that you want to customize and to leave the customizations for the other keys empty. By LISTSERV Maestro’s defaulting mechanism, the original values are correctly combined with your custom texts.

However, if you use the tool for an actual translation to a foreign language, leaving any key value empty must be considered carefully. Some resource keys define values which are actually language-independent or can be used with the default English value even for foreign languages. One example of such a value is the value for the key “`hosted/subs/subscribe.msg.charset.confirmMailForMemberAreaJoin`” in the Maestro User Interface Subscriber Pages component of the translation kit. The value for this key defines a character set and the default value “ISO-8859-1” is suitable for many Western European languages, which means that the custom value for this key can remain empty in many cases. However, most key values are actual texts that require a translation, so if you leave a key value empty for a non-English

translation, the default English text for this key is shown together with the other, translated values, i.e. this results in an undesired language mix-up.

To help you distinguish between supplied and un-supplied parts, each supplied key is marked with a green checkmark. This green checkmark is propagated to higher levels of the resource key tree display once all entries in the next lower level are marked as "supplied completely".

2.7.2 HTML Code

LISTSERV Maestro uses a web-based user interface which consists of HTML pages which are displayed in a web browser.

This means, that the largest part of all language dependent texts is displayed as part of a HTML page and therefore also must follow the rules imposed on HTML.

2.7.2.1 HTML Escape Sequences

In HTML, some characters have special meanings. For example the "<" and ">" characters are used to enclosed special HTML tags which are used to format the HTML page. Such a HTML tag may look like this:

```
<tagname>
```

The appearance of a word enclosed in "<" and ">" signals to the browser that this word is not supposed to appear as a visible word, but that instead is a special HTML tag which conveys some special meaning to the browser, for example how to render and format the following text.

As a consequence, if you actually want a "<" or ">" character to appear on the page, then you need to "escape" this character in form of a HTML escape sequence. Another typical character which needs to be escaped in form of a HTML escape sequence is the ampersand character "&":

'<' must be represented by the escape sequence "`<`"

'>' must be represented by the escape sequence "`>`"

'&' must be represented by the escape sequence "`&`"

'"' must be represented by the escape sequence "`&quot;`"

A non-breaking, non-whitespace space may be escaped as "` `"
(see below, chapter about HTML whitespace).

Note: HTML escape sequences always start with an (un-escaped) ampersand and end with a semicolon. Please see the many related resources on the web that tell you more about characters that need to be escaped in HTML, for example here:
http://www.w3.org/TR/html401/sgml/entities.html.

2.7.2.2 HTML Whitespace and Linebreaks

Whitespace in HTML is interpreted specially: Any sequence of whitespace characters (e.g. space, tab, linebreaks, etc.) will be rendered as a **single** space by the web browser. Example:

```
word 1  \t\t\r\n\t\t\t  \r\nword 2
```

This value starts with "`word 1`" followed by two spaces, two tabs, a Windows-style linebreak, three more tabs, another two spaces, another linebreak and ends with "`word 2`". All the characters between "`word 1`" and "`word 2`", e.g. all the spaces, tabs and linebreaks, are considered whitespace. This means that they will all be collated into a single space when rendered as part of a HTML page: "`word 1 word 2`". There will not be several spaces but only one. And there will not be any visible tab or linebreaks whatsoever.

To actually include more than one space, you need to employ "special" spaces which are written in form of their HTML escape sequence. Widely used is "` `" for a non-breaking space. So

```
word 1   word 2
```

defines the text "`word 1   word 2`" where the three spaces between the two words will actually appear on the rendered page (and, since they are non-breaking spaces, the browser will also not introduce a linebreak at any of these spaces).

Linebreaks are usually introduced by the browser that renders the page whenever a linebreak is required, e.g. whenever the line length is not enough to show the rest of the text, the text will be wrapped to the next line (and the browser will use whitespace between words as the places where a possible linebreak may be inserted for this wrapping).

So linebreaks that are visible on the rendered page cannot be inserted by inserting a linebreak (by pressing the ENTER key) into the translated text. Instead, you need to use a special HTML tag which has the effect of introducing a visible linebreak:

```
<br>
```

Example: The following text

```
line 1<br>line 2
```

will actually appear as two lines "`line 1`" and "`line 2`" on the rendered page.

2.7.2.3 HTML Formatting in Text Resources

The largest part of all text resources in the translation kit are pure text strings, without any formatting directives (in form of HTML tags) in them. All formatting is done on the pages themselves, into which the texts are inserted.

However, some text resources do contain actual HTML tags, which do some sort of formatting that cannot easily be imposed from the outside.

For example, if you have a sentence in which you want to emphasize only a single word in the middle by rendering it in a bold font, you cannot do so with an outside formatting directive, but have to include the corresponding bold-tag right into the text of the sentence itself, bracketing the word you want to emphasize.

If you find a HTML tag (recognizable as being enclosed in "<" and ">") in a text, then you should see to it that the same tag, with the same semantics, appears in your translated version of the text too. Most commonly used are so called span-tags "`<span>`". A span-tag actually consists of an opening part "`<span>`" and a closing part "`</span>`". Those two parts bracket one or several words in the text to which a special formatting shall be applied, for example an emphasis. The kind of formatting is then included in the opening span-part, in form of a class attribute: "`<span class="emphasis">`".

Example:

```
Do you <span class="emphasis">really</span> want to delete this?
```

This defines a text with a question to the user, where the word "really" obviously is to be emphasized, therefore it has been bracketed with the opening and closing parts of the span-tag, using the "`emphasis`" class. So the output will look like this:

> Do you **really** want to delete this?

You should now translate this text to a corresponding meaning in your target language, and employ the opening and closing parts of the span-tag (with the “emphasis” class) to the word that comes closest to the same semantic as the word “really” in the sentence above.

Sample for Swedish:

```
Skall detta <span class="emphasis">verkligen</span> blir raderat?
```

Sample for German:

```
Wollen Sie dies <span class="emphasis">wirklich</span> löschen?
```

The span-tag is frequently used throughout the text resources of LISTSERV Maestro, so look out that your transfer them with a similar meaning to your translation, using the same “class” attribute.

Other tags that are sometimes used are:

`<ul> … </ul>`

`<li>`

`<p> … </p>`

(But this list is not complete).

If you find any HTML tags, be careful to use the same tags, with the same semantic, in your translated version.

2.7.2.4 Non-HTML Exceptions

Even though the bulk of all texts are used inside of HTML pages, there are some exceptions. Inside of these exceptions, the rules of HTML escape sequences, HTML whitespace and linebreaks, and HTML tags, do **not** apply and you must be careful not to for example use a HTML escape sequence in one of these texts. If you would do so, and for example write “`&amp;`” instead of “`&`”, then the user would actually see the text “&” instead of the desired “&”.

The best approach to avoid this is to use HTML markup in translated texts only if the original text contained HTML markup. In many cases, however, the original text does not contain HTML markup but is used inside of an HTML page anyway, like for example the text for the key “`assign.title.page`”, which is used as the title text for the team collaboration page of a LISTSERV Maestro mail job. A translation for this text may benefit from additional HTML markup such as an additional “`<br>`” in the case that the translated version is very long and a linebreak would help achieving a good page layout.

In other cases the text is used in a context where no web browser is involved, e.g. is sent to the user as a plain text mail or is written to a downloaded text file.

2.7.3 Word Replacement

In some resource texts, LISTSERV Maestro employs word replacement to ease translation. Consider the following example:

> “500 duplicates have been removed from the job”.

Obviously, this text contains a value which may vary at run time. Therefore, you cannot simply have a value with this text in the translation, or otherwise you would have to have an infinite number of texts, one for each number of duplicates that may have been removed.

So the immediate solution seems to be, to simply specify the text "*duplicates have been removed from the job*" and let LISTSERV Maestro prepend to this text the number of deleted duplicates at runtime. But that would mean that it would be hard coded that the number of deleted duplicates appears first in the sentence, followed by some text. What if in a different language the syntax of the language would not allow such a sentence? For example, the syntax of that other language may allow only a sentence like "From the job, 500 duplicates have been removed", where the variable part appears right in the middle of the sentence. So obviously, this "hard coded" solution is not a good solution.

Instead, LISTSERV Maestro uses word replacement.

In the text resources, whenever there is a variable part, a replacement placeholder is used instead. At runtime, the placeholder will be replaced with the variable value, forming the desired finished sentence.

2.7.3.1 Standard Numbered Placeholders

Standard numbered placeholders appear in the text in form of a number that is enclosed in curly braces, like this: "`{0}`", "`{1}`", "`{2}`", etc.

So the example from above, we could write as the following text:

```
{0} duplicates have been removed from the job
```

When at runtime, the "`{0}`" is replaced by the variable number "500", the desired correct English sentence "*500 duplicates have been removed from the job*" will be formed. And if in another language the other form of the sentence would be required, the translator can simply "move around" the placeholder in the sentence, for example forming a text like:

```
From the job, {0} duplicates have been removed
```

Here too, after replacement of "`{0}`", the correct sentence would be formed.

This means, that whenever you find one of these numbered placeholders in the text string, recognizable by their curly braces, you must make sure to:

- Include exactly the same number of placeholders in your translated version, with exactly the same number-indexes.
- Determine the meaning of each placeholder (e.g. what will it be replaced with), so that you can use the same placeholder with the same meaning even in your translated version.
- Use the correct order of placeholders, which may vary from the order in the original text (see next chapter).

Numbered placeholders can occur in the text in any sequence, i.e. the first placeholder in a text (when read from left to right) does not necessarily have to be the one numbered "`{0}`".

In most cases this may very well be so, but it is not a general rule that needs to be followed, e.g.

```
This {1} contains {0} two placeholders
```

is just as valid as

```
This {0} contains {1} two placehodlers
```

would be.

More important than the actual 0, 1, 2,… order is, that the semantically correct placeholder is used at each position.

Consider the following sentence: “250 recipients have opened mail XYZ”. This sentence obviously contains two variable values: The number of recipients that have opened the mail (“250”), and the ID of the mail (“XYZ”). Now in another language, the syntax may require this sentence to be written as “The mail XYZ was opened by 250 recipients”. Since we are aware of this, we write the original value with placeholders:

```
{0} recipients have opened mail {1}
```

This means that “`{0}`” will be replaced with the number of recipients and “`{1}`” will be replaced with the ID of the mail. Therefore, in the “other language” version, we would have to make sure to use the correct placeholders in the right positions:

```
The mail {1} was opened by {0} recipients
```

which effectively reverses the order of the two placeholders (when read from left to right). This is however desired, because if we would not reverse the order, and use a text like “`The mail {0} was openend by {1} recipients`”, then after replacement we would get a nonsense sentence like “The mail 250 was opened by XYZ recipients”.

So whenever there are more than one replacement placeholders in the same value, you need to make extra sure to understand the meaning of each placeholder (e.g. what will it be replaced with), so that you can then use each placeholder at the correct position in your translated text, even if that should change the order of the placeholders!

It is also legal for one and the same placeholder to appear several times in the same text. If this is the case, then each occurrence of the placeholder is simply replaced with the same value. The following is an example where the number of selected jobs is merged into two locations of the same text, to create a meaningful message:

```
You have selected {0} mail jobs for deletion.
Are you sure that you want to delete these {0} jobs?
```

2.7.3.2 Named Placeholders on the LISTSERV Maestro User Interface Subscriber Pages

The placeholders on the subscriber pages of the Maestro User Interface in some cases require advanced additional attributes and are therefore written as follows:

```
{{maestro:[placeholder name] [placeholder attributes]}}
```

The placeholder name conveys the semantics of the placeholder in a self-explanatory way, for example the placeholder `{{maestro:subscriberAddress}}` is replaced with the e-mail address of a subscriber navigating the subscriber pages.

The placeholder attributes are written as a sequence of key/value pairs and are only needed for advanced placeholders such as for example `{{maestro:passwordLink text="Click here"}}`, which is replaced with a link to the Request Password page that is rendered with the text “Click here”.

2.7.3.3 Placeholder Validation

Each text that you supply as a translated version for a resource key is checked to contain exactly the placeholders (numbered or named) that are present in the original text. Take for example the original text for the key “`hosted/subs/login.title.login`” in the LISTSERV Maestro User Interface Subscriber Pages component of the translation kit:

```
Login for {{maestro:datasetName}}
```

The supplied translated text for this key is validated to contain the text "`{{maestro:datasetName}}`". Any other string in the form of `{{maestro:myPlaceholder}}` or `{0}` (for any number) will be denied as "placeholder unknown in original text".

Similarly, consider the original text for the key "`newJobs/authorizeSending/label/clickOnNumberOfLinks`" in the LISTSERV Maestro User Interface Main Application component of the translation kit:

```
Click on at least one of {0} links
```

This original text defines that the placeholder "`{0}`" is known and supported, any other occurrence of a string that looks like a numbered or a named placeholder is denied upon validation as "placeholder unknown in original text".

2.7.4 Miscellaneous

2.7.4.1 Date and Time Formats

Date formats, e.g. the order in which day, month and year appear when printed as a date, and the separator character between them, differ from country to country. While Americans usually write "`month/day/year`", the Swedes write "`year-month-day`" instead, and Germans use "`day.month.year`".

The user can choose which output and input formats for date and time he prefers, by setting them as his preferences in the LISTSERV Maestro UI, in the menu "User Settings → Preferences".

While the user has not chosen his own preferences, dates and times are printed and parsed using default settings.

These default settings can be influenced during translation, so that the defaults match the target language you are translating to. E.g. all users without individual preferences set will then see dates and times formatted in a way that matches your target language.

For this, there are some special date format keys which you need to take into special consideration:

(The remainder of this paragraph assumes that you are working on the LISTSERV User Interface Main Application Resources. All nodes mentioned here can be found in this translation kit component.)

- **Output Format for Date**

  Whenever LISTSERV Maestro outputs dates, for example the send-date of a job, you have two attributes of the output format that you can influence:

  - The name of the month:

    Locate the node "`date`". Underneath this node, edit the twelve keys called "`date.month.1`" to "`date.month.12`". Assign values to these twelve keys that match the name of the months in your target language (starting with January), preferably in an abbreviated textual form. If there is no good abbreviated textual form in your target language, and the full textual form appears as too long, you may also use numerical values 1 to 12 instead. (The names of the months are independent of any preferences the user sets, e.g. they are always used for date output.)

- Print month or day first:

  Locate the node "`date`". Underneath this node, edit the key called "`date.printMonthFirst`". Set the value to "`true`" to get an output date format of the type "Month Day, Year" (e.g. "American" style, as in "Aug. 12, 2002") or to "`false`" to get a format of the type "Day Month Year" (e.g. "European Style", as in "12 Aug. 2002"). (This value defines the default for users without an individual preference setting.)

- **Output Format for Time**

  Whenever LISTSERV Maestro outputs times, for example the send-time of a job, you have two attributes of the output format that you can influence:

  - Use AM/PM format or not:

    Locate the node "`date`". Underneath this node, edit the key called "`date.useAM/PM`". Set the value to "`true`" if you want to use a 12 hour format with am/pm or to "`false`" if you want to use a 24 hour format (military time). (This value defines the default for users without an individual preference setting.)

  - Separator character for hours and minutes:

    Locate the node "`date`". Underneath this node, edit the key called "`date.timeSeparator`". Set the value to the character that is to be used to separate hours and minutes. For example "`:`" if the desired output is something like "09:15". (This value defines the default for users without an individual preference setting.)

- **Report Download Format for Date and Time**

  Whenever LISTSERV Maestro prepares report results for download, in form of a ZIP file, it puts a "`readme.txt`" text file into that ZIP file with some information about the downloaded report. This information also contains some date and time information, for example about when the report was executed.

  To set the appearance of this date/time format locate the node "`reports`" and edit its sub key "`reports.msg.downloadDateFormat`". Set its value to a pattern that will be translated into an actual date and time string at runtime. In this pattern, the following pattern fields **must** appear, but you may juggle them around, change their order and insert other fill characters, to create a format that suits you. Note that uppercase and lowercase are important, and that all pattern fields will be replaced with numerical counterparts (e.g. no textual names of months, etc.).

  - `MM`: Two uppercase M-characters: Will be replaced with the month of the year 01-12.
  - `dd`: Two lowercase d-characters: Will be replaced with the day of the month 01-31.
  - `yyyy`: Four lowercase y-characters: Will be replaced with the 4-digit value of the year.
  - `HH`: Two uppercase H-characters: Will be replaced with the hour of the day 00-23.
  - `mm`: Two lowercase m-characters: Will be replaced with the minute of the hour 00-59.

Examples: 28th of August 2002, 5 minutes after 3, p.m., will be formatted as follows:

`MM/dd/yyyy HH:mm` → 08/28/2002 15:05

`yyyy-MM-dd HH:mm` → 2002-28-08 15:05

`HH:mm [dd.MM.yyyy]` → 15:05 [28.08.2002]

(These settings are independent of any user preferences, e.g. the user can not influence the date and time formats of these downloaded reports.)

- **Input Format for Date and Time**

  Whenever the user needs to input a date and time value for LISTSERV Maestro to understand, the user must follow a certain pattern (which is also displayed to the user). You can influence this pattern:

  - Date pattern:

    To set the input date format, locate the node "`date`"and edit its sub key "`date.dateFormat.default`". Set it to a value that is either the numerical pattern number of any of the predefined date patterns (e.g. "`1`", "`2`", "`3`" or "`4`", see below), or set it to "`custom`" if you want to define a custom date pattern for input (see below). (This value defines the default for users without an individual preference setting.)

    You can choose between four pre-defined date patterns, or provide your own customized pattern. Each pattern exists as two versions: The pattern used for parsing the user input, and the pattern that is displayed to the user, so that the user knows which format to use for the input (e.g. this "display pattern" is simply a string which is displayed in form of a small help text, to help the user find the right formatting for the date he has to enter).

    The predefined patterns are:

Pattern Number:	Display Version:	Parse Version:
1	`mm/dd/yyyy`	`MM/dd/yyyy`
2	`dd.mm.yyyy`	`dd.MM.yyyy`
3	`dd/mm/yyyy`	`dd/MM/yyyy`
4	`yyyy-mm-dd`	`yyyy-MM-dd`

    The "parse version" of the predefined patterns cannot be changed. However, you may change the "display version" of the predefined patterns: Open the LISTSERV User Interface Main Application Resources, locate the node "`date`"and edit the keys called "`date.dateFormat.N.forDisplay`", where "`N`" stands for the pattern number of the pattern in question.

    (The current display versions of the predefined patterns reflect the exact structure of the matching parse versions, only that the upper-case letter "`MM`" for "month" is displayed as lower-case "`mm`", as to not confuse the user too much.)

    If none of the predefined patterns matches your requirements, you may define a custom pattern. In that case, you must supply **both**, the display version and the parse version of that pattern.

    For the display version, you are free to choose what it looks like (however, you should of course choose a pattern that makes it easy for the user to understand the formatting that is required).

To define the display version, edit key called "`date.dateFormat.custom.forDisplay`" (this value is initially empty).

For the parse version, you need to follow strict rules: The pattern string must contain exactly the **three** pattern fields "`MM`", "`dd`" and "`yyyy`" (with that upper and lower case). See above in "Report Download Format for Date and Time" for details of these three fields and examples. You can order them in any way you like and include any separator or fill characters you like, but you must be aware that the user must enter them in exactly the same order with the same separators and fill characters, so that LISTSERV Maestro can understand them. Therefore you should probably choose an order and separators and fill characters which are most common in the locale matching your target language (and should also provide a similar display version, with the same separators and fill characters, see above).

To define the parse version, edit key called "`date.dateFormat.custom.forParsing`" (this value is initially empty).

- Time pattern:

To set the input time format, locate the node "`date`" and edit the key "`date.timeFormat.default`". Set it to a value that is either the numerical pattern number of any of the predefined time patterns (e.g. "`1`" or "`2`", see below), or set it to "`custom`" if you want to define a custom time pattern for input (see below). (This value defines the default for users without an individual preference setting.)

You can choose between two pre-defined time patterns, or provide your own customized pattern. Each pattern exists as two versions: The pattern used for parsing the user input, and the pattern that is displayed to the user, so that the user knows which format to use for the input (e.g. this "display pattern" is simply a string which is displayed in form of a small help text, to help the user find the right formatting for the date he has to enter).

The predefined patterns are:

Pattern Number:	Display Version:	Parse Version:
1	`hh:mm`	`HH:mm`
2	`hh.mm`	`HH.mm`

The "parse version" of the predefined patterns cannot be changed. However, you may change the "display version" of the predefined patterns: Locate the node "`date`" and edit the keys called "`date.timeFormat.N.forDisplay`", where "`N`" stands for the pattern number of the pattern in question.

(The current display versions of the predefined patterns reflect the exact structure of the matching parse versions, only that the upper-case letter "`HH`" for "hour" is displayed as lower-case "`hh`", as to not confuse the user too much.)

If none of the predefined patterns matches your requirements, you may define a custom pattern. In that case, you must supply **both**, the display version and the parse version of that pattern.

For the display version, you are free to choose what it looks like (however, you should of course choose a pattern that makes it easy for the user to understand the formatting that is required).

To define the display version, locate the node "`date`" and edit the key called "`date.timeFormat.custom.forDisplay`" (this value is initially empty).

For the parse version, you need to follow strict rules: The pattern string must contain exactly the **two** pattern fields "`HH`" and "`mm`" (with that upper and lower case). See above in "Report Download Format for Date and Time" for details of these three fields and examples. You can order them in any way you like and include any separator or fill characters you like, but you must be aware that the user must enter them in exactly the same order with the same separators and fill characters, so that LISTSERV Maestro can understand them. Therefore you should probably choose an order and separators and fill characters which are most common in the locale matching your target language (and should also provide a similar display version, with the same separators and fill characters, see above).

To define the parse version, locate the node "`date`" and edit the key called "`date.timeFormat.custom.forParsing`" (this value is initially empty).