Structure Tools 2.30
User Guide

Table of Contents

Chapter 1 — Introduction

Thank you for using or evaluating the Structure Tools plugin for structured FrameMaker. If you have any comments or questions regarding the software or its documentation, please send them to info@weststreetconsulting.com.

Important disclaimer

When you use Structure Tools, you agree to do so at your own risk. West Street Consulting and affiliates are not responsible for any damages or loss to software, hardware, or data, whether by user or software error, or due to errors in this documentation.

What is the Structure Tools plugin?

Structure tools provides a number of convenience functions for the structured FrameMaker interface. Specifically, it provides

- **Enhanced bookwide processing** With the comprehensive book processing feature, you can prepare your book for publishing from one central dialog box, including a book update, setting conditional text, hiding element boundaries, and automatic structuring of generatable files. For more information, see “Book processing with Structure Tools” on page 5.
- **Automatic attribute population and improved defaults handling** With Structure Tools, you can set up customizable values to automatically populate attributes when you insert and wrap elements. These values can include “dynamic” values such as file names and the current date. In addition, Structure Tools provides features to enhance the current EDD method of attribute defaults handling. For more information, see “Attribute auto-population” on page 16.
- **Custom attribute editors** Structure Tools includes two custom attribute editor which enhances attribute editing capabilities, including an expanded “choices” feature. For more information, see “Custom modal attribute editor” on page 20 and “Custom modeless attribute editor” on page 21.
- **Automatic window configuration (FrameMaker 7.x and 8.x only)** With Structure Tools, you can customize a “workspace” of sorts, including document positioning, zoom, and symbol display. With this feature, you can store your preferred settings for document and Structure View windows, and can then reconfigure one or all with a click. For more information, see “Automatic window configuration” on page 9.
- **General shortcuts** Structure Tools provides a number of quick shortcuts for common structured interface activities, such as toggling element brackets and refreshing EDD definitions. For more information, see “General shortcuts and features” on page 12.

Overall vision and purpose

The Structure Tools plugin is intended as a general enhancement for structured FrameMaker usage. For the most part, it serves to provide automation for a variety of common activities which would otherwise cause a greater expense of time.

Structure Tools is based on the author's best judgment of most-needed features, gathered from personal usage and input from users like you. If you have suggestions for Structure Tools, please send them to West Street (info@weststreetconsulting.com).

Trademarks

Chapter 2 — Book Processing and General Features

This chapter contains information on the consolidated book processing feature of Structure Tools, and other general-purpose features of the plugin.

Book processing with Structure Tools

The special book processing feature of Structure Tools (**WS Structure Tools > Process Book**) consolidates the following book-wide activities into a central access point:

- Book update *(see page 7)*
- Structuring (restructuring) generatable files *(see page 8)*
- Hiding element boundaries
- Managing conditional text settings
- Saving files

The book processing feature can serve as a general-purpose tool for managing your book. In addition, you may find that it serves as a final step for preparing a book for publication, such as exporting to PDF or processing by a help-generation program such as Quadralay® ePublisher®.

The **Process Book** dialog box includes the following options:

Book update area — Sets options for a FrameMaker book update, identical to a normal book update, with some exceptions. For more information, see “Book update” on page 7.

Note: If you choose to restructure generatable files, the location of the conversion table document must be correctly specified in your general settings. For more information, see “General settings” on page 10.

Allow Structure Tools to open files — Allows the plugin to attempt to open any book components that are currently closed. If any components are currently closed and this option is not selected, book processing cannot proceed.

Note: With FrameMaker 9, there is a delay when opening the first file in a workspace as the pods and layout regenerate. If you allow Structure Tools to process a book that is completely closed and no other files are currently open, the book processing can take some time, especially if you have selected **Save and close any files that were opened**.

Save and close any files that were opened	Causes Structure Tools to save and close any files that it had to open, after processing them. If this option is selected, each file is saved and closed individually as soon as its respective processing is complete. If this option is not selected, any files that were opened remain open after processing is complete. ***Note:*** Before selecting this option, you should be absolutely sure that you trust the results of the book processing. All changes will be saved and you will not be able to recover the original files.
Display warning when closed files are detected	Causes Structure Tools to prompt for confirmation when it detects book components that are not open. This is a recommended setting.
Hide element boundaries	Hides all element brackets and tags. If unchecked, current boundary display is unaffected.
Refresh element definitions	“Reimports” each document’s internal EDD, causing a refresh of element definitions and the removal of format overrides. This option is identical to the “Refresh EDD” function described in “General shortcuts and features” on page 12.
Save all files, including book	Following all processing, saves all files and the book. If unchecked, *no files are saved afterwards.* ***Note:*** Before selecting this option, you should be absolutely sure that you trust the results of the book processing. All changes will be saved and you will not be able to recover the original files.
Hide condition indicators	Hides all conditional text indicators in all files. If unchecked, current indicator status is unaffected.

Show/hide conditions	Allows you to specify conditional text settings for all files. If unchecked, current conditional text settings are unaffected. If a given condition does not exist in a particular file, the setting has no effect for that file.
FrameScript area	Allows you to specify FrameScripts to run before and/or after the internal processing sequence. Note the following: • You must have FrameScript installed and its name properly registered with Structure Tools in your general settings. For more information, see “General settings” on page 10. • Structure Tools calls the “`Event NoteClientCall`” event handler in the script. Therefore, you must have this set up in the script. For more information on how scripts respond to FDK calls, see the FrameScript documentation. • The “pre-processing” script runs before Structure Tools does anything related to its own book processing sequence. The “post-processing” script runs after Structure Tools has completed the entire book processing sequence *except* saving files, if that option was enabled. For more information, see “Processing order” on page 7. • To prevent FrameScripts from running, leave these fields blank.

Processing order

During book processing, the individual events follow this sequence, as applicable:

1. Run the “pre-processing” FrameScript
2. Remove current content from structured generatable files that will be regenerated
3. Hide element boundaries
4. Show/hide conditional text
5. Hide condition indicators
6. Refresh element definitions
7. Perform book update
8. Restructure generatable files, as applicable
9. Run the “post-processing” FrameScript
10. Save all files

Book update

The book update performed by Structure Tools is identical to a normal book update, with the exceptions noted in the following sections.

Structuring (restructuring) generatable files

With a normal book update, generatable files (such as TOCs, indexes, and lists) must be unstructured. The generation process uses paragraph formats as the foundation for building the files, and has no knowledge of structure definitions or desired hierarchy. If you attempt to regenerate a structured generatable file, the results will be unpredictable at best.

If you prefer structured generatable files, Structure Tools overcomes this limitation by automatically invoking the structure generator following book processing, to restructure the files using the conversion table process. As an example, assume that you have a structured TOC, and want to update your book with the TOC remaining structured afterwards. Structure Tools can automate the following tasks, which would otherwise be manual:

1. Delete all current content from the TOC, generally by deleting the highest-level element.
2. Run a book update to regenerate the TOC.
3. Open the conversion table and run the structure generator to restructure the TOC.
4. Replace the old TOC with the new file produced by the structure generator.

Therefore, to use this feature, you must meet the following minimum requirements:

- **A valid conversion table** You must have a valid, working conversion table developed. Structure Tools does not make decisions about the structure of your generatable files; rather, it simply invokes the structure generator using your conversion table. All instructions concerning the final architecture of the structure must be contained in the conversion table. For more information on conversion tables and the structure generator, see your *Structure Developers Guide*.
- **A valid path to the conversion table specified in your general settings** The location of the conversion table is stored in your Structure Tools general settings. You can specify the location of this file two ways, either absolute or relative. For more information, see “General settings” on page 10.

  Tip: Structure Tools can automatically find a conversion table, if it resides in the book folder with the generic name `ConversionTable.fm`. In this case, a valid path need not be specified in your general settings. If a valid path is specified, Structure Tools will look there for the table before searching the book folder.

In the Process Book dialog, you have the following options regarding generatable file restructuring:

Skip files that are currently structured	Regenerates all generatable files that are currently unstructured, and leaves them unstructured. Any generatable files that are currently structured are left as-is, ungenerated.
Regenerate all, and leave all unstructured	Regenerates all generatable files in the book, unstructured or not, and leaves all files unstructured at the end. In other words, all files are regenerated and no files are restructured.

Regenerate all, and only restructure files that were structured originally	Regenerates all generatable files in the book, and only restructures those that were structured before the regeneration.
Regenerate all, and restructure all files	Regenerates all generatable files, structured or not, and restructures all afterwards.

Tip: The plugin determines that a file is structured by the existence of a highest-level element in the main flow. Any elements below the HLE, valid, invalid, or non-existent, have no effect. If any HLE exists, the file is considered to be structured.

Note: If a file was structured beforehand, Structure Tools "remembers" the attribute settings of the highest-level element and reapplies them to the HLE of the newly-structured document. In this manner, it preserves attribute settings of HLEs during the structuring process. If your conversion table causes the installation of a different HLE than originally existed, however, the attributes applied to the new HLE may not be valid, depending on differences between the element definitions.

Also, note that structured generatable files are not required for general structured FrameMaker use. This Structure Tools feature is intended for users that have a special need to maintain generatable files in a structured format. The everyday benefits of structured authoring normally do not apply to generatable files, since FrameMaker authors those files automatically.

Attribute-based running header/footer bug workaround

In a structured book, you can base running header/footer variables on attribute values, including attributes in the book structure itself. Some users choose this option to store running H/F values at the book level, which should update throughout all files during a book update.

However, in some versions of FrameMaker, a bug exists in this process concerning generatable files. If you have attribute-based running H/F variables in a generatable file, and they reference attributes at the book level, they may not properly update with a normal book update. With a Structure Tools book update, though, all attribute-based variables should update as expected.

Automatic window configuration

Note: Due to changes in the user interfaces, this feature is not available for FrameMaker 9.

With Structure Tools, you can capture your preferred settings for window configuration, then automatically reconfigure one or all windows to those settings. Captured settings include:

- Position
- Height and width
- Zoom
- Element boundary display
- Text symbol display

- Ruler display
- Border display
- Window maximization state
- Comparable settings for the Structure View window

To capture preferred window settings

With a document window active and configured as desired, select **Element > WS Structure Tools > Capture Preferred Window Settings**.

Note: If the Structure View window is open, settings for it will be captured as well.

To reconfigure an open window or windows

With a document window active and configured as desired, select **Element > WS Structure Tools > Reconfigure Current Window**.

-or-

Select **Element > WS Structure Tools > Reconfigure All Windows**.

To have windows automatically reconfigure upon opening documents

Set the appropriate option in your general settings. For more information, see “General settings” on page 10.

General settings

To set general Structure Tools settings, select **WS Structure Tools > General Settings**. In the General Settings dialog, you can set the following:

Enable attribute auto-population	Globally enables the attribute auto-population feature, controlled by your attribute settings. For more information, see “Attribute auto-population” on page 16.
When pasting attribute values, paste empty values	Allows “empty” attributes to be pasted during an attribute paste operation. For more information, see “Copying and pasting attribute values” on page 22.
Enable custom attribute editor	Enables the custom modal attribute editor. If enabled, you can select which elements should invoke the editor, or which elements should not invoke the editor. The “new element options” option refers to the document-level setting available from the **Element** menu which FrameMaker normally uses to control its own attribute editor.
After setting attrs, show pop. & req. only	For the modal attribute editor only, allows the attribute display in the Structure View to auto-collapse after values are set. This setting does not affect the modeless editor, which has its own set of options to control this behavior.

When documents are opened, reconfigure windows

Not applicable to FrameMaker 9

Sets Structure Tools to automatically reconfigure document windows when you open the files. Reconfiguring is performed according to your stored settings. For more information, see “Automatic window configuration” on page 9.

When reconfiguring documents, include Structure View

Not applicable to FrameMaker 9

Includes Structure View reconfiguration any time a document window is reconfigured.

When importing XML, correct table title positions

Automates the “fix table titles” feature during XML import. For more information, see “General shortcuts and features” on page 12.

Main menu location

Controls where the **WS Structure Tools** menu appears, either on the main FrameMaker menu or under the **Element** menu.

Conversion table location

The location of the conversion table for generatable file restructuring, during a Structure Tools book process. For more information on book processing, see “Book processing with Structure Tools” on page 5.

This specification can be an absolute path, such as:

```
C:\Data\MyConversionTable.fm
```

...or just a file name, such as:

```
MyConversionTable.fm
```

If only a file name is specified, the Process Book routine will look for the file in the same folder as the book.

Notes: If a full path is specified, but the file cannot be found, Structure Tools will still attempt to find the file in the book folder, using the same filename.

In addition, if no path is specified or the file cannot be found anywhere, Structure Tools will look for a generic `ConversionTable.fm` file in the book folder. In this case, the file need only reside in the book folder with this generic name, and a valid path need not be specified in your general settings.

FrameScript installed name — The name of FrameScript exactly as it is registered in `maker.ini`. This setting is only relevant if you intend to run FrameScripts as a part of the book processing function. For more information, see “Book processing with Structure Tools” on page 5.

Note: FrameScript functionality is currently not implemented, so this field is reserved for future use.

FrameMaker main title bar label — An optional, customizable label for the main title bar of the FrameMaker application. This setting has no effect on FrameMaker 9.x or later. FrameMaker must be restarted for any changes to take effect.

Ctrl+A behavior features

Optionally, you can set the plugin to respond in one of the following ways when the standard **Ctrl+A** shortcut is invoked:

- Select the entire contents of element containing the insertion point. Then, upon a subsequent invocation, revert to standard “select all” behavior.

  -or-

- With each invocation, “walk” the selection up the structure tree until reaching the highest-level element. With this option, standard “select all” behavior is completely disabled while the insertion point is within a structured flow.

This setting can be configured in the “additional settings file” (**WS Structure Tools > Open Additional Settings File**). See the contents of that file for more information.

Autonomous attribute actions

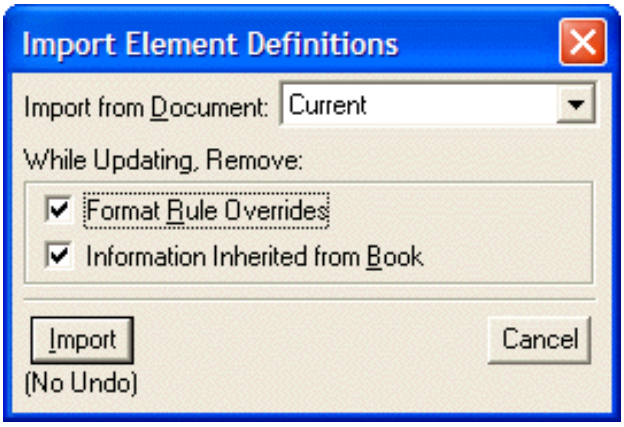
Optionally, you can set the plugin to automatically toggle the attribute display to only populated and required attributes upon key user actions, such as opening and saving documents. These settings can be configured in the “additional settings file” (**WS Structure Tools > Open Additional Settings File**). See the contents of that file for more information.

General shortcuts and features

Structure tools includes a number of general shortcuts, including:

- **Expand/Collapse elements and attribute displays** In the Structure View, the right-click menu includes a variety of Structure Tools commands to expand and collapse elements in different ways. The “generic” Expand Elements and Collapse Elements commands do a full expand/collapse of the selected element, differing from the native “+” and “-” icons on the element bubbles in that they expand/collapse all descendants as well. Several other commands allow more specific expand/collapse behavior and attribute display configuration.

- **Expand Tree To Current Element** This command is found in the right-click menu over document text and expands the Structure View tree directly to the selected element, or the element containing the insertion point. The Structure View will be opened automatically if it is currently closed.
- **Toggle element brackets (Esc q q)** Toggles element brackets on and off.
- **Refresh EDD (Esc q w)** Refreshes the current element definitions of the document. This command performs the same function as selecting File > Import > Element Definitions, selecting Current and all options, and clicking Import.



- **Bring book to front (Esc b b)** Brings the open book window to the front. If multiple book windows are open, it toggles between them. This command only appears for FrameMaker 8 and earlier.
- **Fix Table Titles** This command provides a workaround for a current FrameMaker bug (some FrameMaker versions) associated with tables and XML import. According to the FrameMaker table structure model, the table title element must always appear as the first component in the table hierarchy. This model is true whether the applied table format visually renders the title above or below the table. Therefore, all tables must be structured as such in XML for a proper import into FrameMaker, after which the applied table format should take over and visually position the title correctly.

  In reality, however, FrameMaker ignores the title position setting in the applied format and always puts the table title above the table. That is, even if the format specifies that the title should appear below the table, FrameMaker will override this setting and put the title above. As such, the Fix Table Titles command is designed as a post-processing step after XML import to fix this problem. When run on a document, it looks for any table whose title appears above, but whose table format specifies the title below, and resets that table to conform to the table format with the title below.

  This is a highly specialized command for a very specific issue regarding XML import. You should not use it unless you are absolutely sure you know what it is doing and that it applies to your situation.

Note: The shortcuts for these commands are customizable in the “additional settings file” (**WS Structure Tools > Open Additional Settings File**).

Chapter 3 — Attribute Tools

Structure Tools has a number of features focused on structural attributes. The features fall into two primary categories:

- Attribute auto-population *(see page 16)*
- Custom attribute editors, including:
  - Custom modeless attribute editor *(see page 21)*
  - Custom modal attribute editor *(see page 20)*

In addition, Structure Tools has a central point for attribute settings, which includes parameters that affect both of these feature categories (see “Attribute settings” on page 15).

Note: Structure Tools generally acts upon attributes during standard user actions such as inserting and wrapping elements. However, due to a FrameMaker limitation, the plugin is not able to detect when a default element is inserted by pressing the Return key in the document window. Therefore, auto-population and the custom editor will not work in this instance. Both will work, however, if you follow through with the insertion and then double-click the attributes in the Structure View.

Attribute settings

The Structure Tools attribute settings (**WS Structure Tools > Attribute Settings**) are a central repository of parameters which affect auto-population and the custom attribute editor. This section provides a brief description of each setting, but you may want to follow the links provided to get more information and a better context of how these settings fit into the overall functionality.

In the **Attribute Settings** dialog, you can edit the following:

Attribute names	Attributes that will have settings for consideration by Structure Tools. If an attribute is not listed, Structure Tools functionality will have no effect on it. An unlisted attribute will, however, still appear in the custom attribute editor, if that editor is used.
Attribute settings are active	Activates the attribute settings for the selected attribute. If unchecked, the settings for that attribute will have no effect on Structure Tools functionality, but the settings will remain stored so that you can reactivate them with this checkbox.
Applicable elements	Indicates to which element(s) the settings apply. For the selected attribute, the settings will only apply to that attribute the specified element(s). To indicate all elements, specify **(All elements)**. Note that you can put the same attribute in the list more than once, with different applicable elements. This allows you to specify different behavior for the same attribute on different elements.

Auto-population area

If EDD specifies defaults, make them "real"	Causes Structure Tools to retrieve default attribute values from the EDD, if they exist, and physically populate them as actual values. This feature includes two options: • **Add defaults to values below** If you have additional auto-populate values specified, Structure Tools will combine them with the EDD-specified defaults and populate the attribute with all. • **Replace values below with defaults** If you have additional auto-populate values specified, they will be ignored and replaced by the EDD-specified defaults. For more information, see "Attribute auto-population and EDD defaults" on page 19.
Values to auto-populate	The values to auto-populate for the selected attribute. This list may include static values and "dynamic" values. For detailed information on how dynamic values work, see ""Special" auto-population values" on page 17. ***Tip:*** You can double-click an existing item in the scroll box to edit it.

Custom attribute editor area

Restrict attribute from custom Structure Tools attribute editor	Prevents the attribute from appearing in the custom Structure Tools editor, making it essentially non-editable if you use the custom editor. It does not affect the native FrameMaker attribute editor.
Values to offer as choices	List of values to populate in the "choices" list, in the custom modal attribute editor, or the "available values" list in the modeless editor. This list can contain static values with or without building blocks for creating dynamic values. For detailed information on how dynamic values work, see ""Special" auto-population values" on page 17. ***Tip:*** You can double-click an existing item in the scroll box to edit it.

Attribute auto-population

When you insert an element, wrap an element, or edit attributes, Structure Tools can automatically populate attribute values based on your attribute settings. For more information on attribute settings, see "Attribute settings" on page 15.

Note: For auto-population to function, it must be globally enabled in your general settings. For more information, see "General settings" on page 10.

The behavior of auto-population differs according to how you use the native attribute editor, the custom modal attribute editor, or neither, as follows:

- **No attribute editor** If you use no attribute editor during an element action, the auto-populate values will simply appear on their own. For example, if you insert an element, and both the native and custom editors are turned off, the inserted element will appear with the applicable attributes auto-populated.
- **Custom attribute editor** If you are using the custom attribute editor, auto-populate values will appear in the custom editor first, then appear on the element when you click OK. You will have the opportunity to edit them in the custom editor before you click OK, as with any attribute and value.
- **Native attribute editor** If you are using the native attribute editor, auto-populate values will only appear once the editor is dismissed, and any conflicting specification in the native editor will be overridden. That is, the auto-populate values will not appear in the native editor, but will override anything specified there. For example, assume you have the attribute `MyAttr` set to auto-populate with “MyValue,” and you are inserting an element with the `MyAttr` attribute. When the native editor appears, it will list *<no value>* for `MyAttr`. After you dismiss the editor, “MyValue” will appear on the element, overriding anything you may have specified in the native attribute editor.

In general, Structure Tools attribute settings are designed to work with the custom editor, and you may find that simply turning off the native editor will give you the best results. Use of the native editor with auto-population turned on may produce some unexpected behavior.

Auto-population settings

All values and associated parameters for auto-population are specified in the Structure Tools attribute settings. For more information, see “Attribute settings” on page 15.

Note: For auto-population to function, it must be globally enabled in your general settings. For more information, see “General settings” on page 10.

“Special” auto-population values

Note: This functionality has changed from version 1.x. It has become more flexible, but it will require you to configure the newer version differently than the older version.

In addition to static values, you can use building blocks to create dynamic attribute values, mostly focused on creating values based on filenames, dates, and or times. The following table describes the supported building blocks, many of which are identical to the building blocks offered by FrameMaker for creating system variables:

Building block	Result
<$doc_filename_only>	The filename of the active document. For example: **`MyDoc.fm`**
<$doc_path_only>	The pathname only (no filename) of the active document. For example: **`C:\MyDocuments\FM_Project`**
<$doc_path_and_filename>	The fully-qualified path to the active document. For example: **`C:\MyDocuments\FM_Project\MyDoc.fm`**

Building block	Result
<$book_filename_only>** **<$book_path_only> **<$book_path_and_filename>**	Identical to the “doc” counterparts, except using the path of the book containing the document. The desired book must be open. If Structure Tools cannot find an applicable, open book, the auto-population will fail and the value will be include an error message.
<$year>	The current year, for example: `2009`
<$shortyear>	The abbreviated current year, for example: `09`
<$monthnum>	The current month number without leading zeros, for example: `8` (August)
<$monthnum01>	The current month number with leading zeros, for example: `08` (August)
<$monthname>	The current month name, for example: `August`
<$shortmonthname>	The current abbreviated month name, for example: `Aug`
<$daynum>	The current day date without leading zeros, for example: `3` (third day of the month)
<$daynum01>	The current day date with leading zeros, for example: `03` (third day of the month)
<$dayname>	The current day name, for example: `Tuesday`
<$shortdayname>	The current abbreviated day name, for example: `Tue`
<$hour>	The current hour without leading zeros, for example: `1`
<$hour01>	The current hour with leading zeros, for example: `01`
<$hour24>	The current hour in 24-hour time, with leading zeros if applicable, for example: `01` and `13`
<$minute>	The current minute without leading zeros, for example: `1`
<$minute01>	The current minute with leading zeros, for example: `01`
<$second>	The current second without leading zeros, for example: `1`
<$second01>	The current second with leading zeros, for example: `01`
<$ampm>	The current a.m. vs. p.m. condition, lowercase without periods, for example, `am`

Building block	Result
<$a.m.p.m.>	The current a.m. vs. p.m. condition, lowercase with periods, for example, `a.m.`
<$AMPM>	The current a.m. vs. p.m. condition, uppercase without periods, for example, `AM`
<$A.M.P.M.>	The current a.m. vs. p.m. condition, uppercase with periods, for example, `P.M.`
<$var[*varname*]	The current value of the variable specified for *varname*. For example, if you had a variable called "MyVar" in a document, you could use the following auto-population value: **<$var[MyVar]>** When the applicable attribute is auto-populated, the value will be derived from the current value of the "MyVar" variable. ***Note:*** Variable text is drawn directly from the variable format definition, and any building blocks will be displayed literally and unprocessed.

The syntax for these values must be exact. In the attribute settings, when you click "Add" to add an auto-populate value, the drop-down list in the following dialog is provides several examples for convenience.

Attribute auto-population and EDD defaults

In an EDD, you can specify one or more default values for certain types of attributes. EDD-specified defaults appear in the Structure View as italicized values, but do not appear in the attribute editor for the respective elements. That is, if an EDD-specified default appears in italics in the Structure View, and you launch the native attribute editor for that element, the respective attribute will still appear in the editor as *<no value>*.

In your attribute settings, for any particular attribute, you can choose to make EDD-specified defaults "real" during auto-population actions. In essence, this function causes any EDD-specified defaults to populate the attribute as genuine values, appearing in normal font in the Structure View.

The behavior of making defaults "real" is virtually identical to other auto-population actions, except that the values to auto-populate are drawn directly from the EDD. With all other types of auto-population, all information comes from the Structure Tools attribute settings, and no interaction with the EDD takes place.

Auto-population and existing elements

Auto-population remains active when you launch an attribute editor for an existing element, as follows:

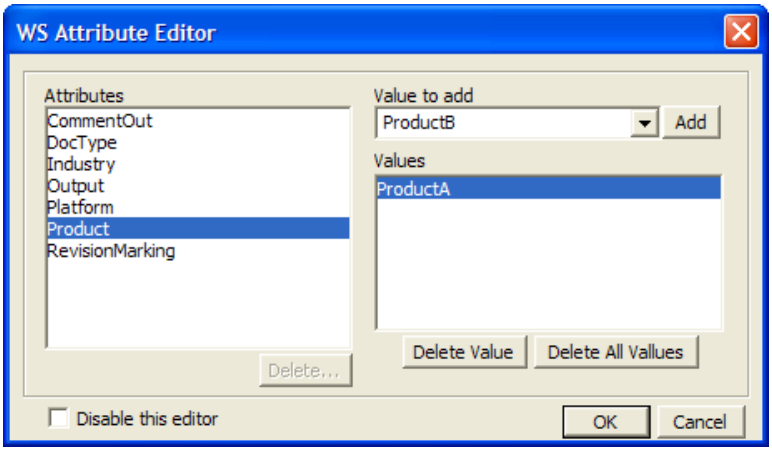
- If you have auto-population parameters set for a particular attribute, and an actual attribute is currently empty, it will be auto-populated before the editor launches (native

or custom editor). The new values will appear in the editor, and you can simply click “Done” or “OK,” as applicable, to set those values.

- If an attribute currently has one or more values, it is unaffected by auto-population parameters. Auto-population never alters an attribute on an existing element that has existing values.

Custom modal attribute editor

Structure tools includes a custom modal attribute editor, designed to replace the native modal editor that appears when you insert/wrap elements and double-click attributes in the Structure View. It is intended to improve editing features over the native editor, working in conjunction with the defaults-handling and “choices” settings in your attribute settings “Attribute settings” on page 15.



The functionality of the editor should be reasonably intuitive, noting the following:

- **The editor must be specifically enabled** Before the editor will appear, you must enable it in your general settings. For more information, see “General settings” on page 10.
- **For those elements that invoke it, the custom editor replaces the native editor** Depending on your general settings, you can use both editors during a session, but you should never see both editors appear for any given element.
- **The custom editor is launched in the same manner as the native editor** It will appear automatically after inserting or wrapping an element, as applicable, and when you double-click attributes in the Structure View.
- **No “Set Value” button** Unlike the native editor, you do not need to click a button such as “Set Value” to set values before clicking OK, or switching to another attribute. Once you add a value with the Add button, it is considered “set,” unless you click Cancel afterwards.
- **Enhanced “choices” handling** The drop-down list in the editor contains all EDD-specified choices and/or all choices specified in your attribute settings. For more information, see “Attribute settings” on page 15.
- **Auto-populated values are preset** All auto-population is done before the custom editor launches, so those values will appear preset in the editor. You can, of course, choose to edit them afterwards as desired. Note that this behavior is not true for the

native editor, which does not show auto-populated values during the initial insertion or wrapping of an element.

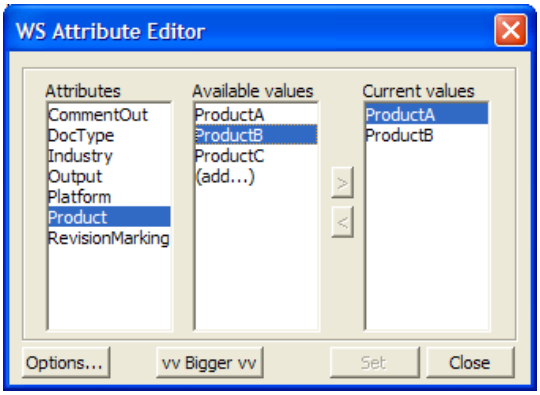
- **No hidden or read-only attributes** Unlike the native editor, the custom editor does not list read-only attributes at all. If you attempt to invoke the editor on an element that has no editable attributes, the editor will simply not appear. In addition, you can set individual attributes to be restricted from the custom editor. For more information, see "Attribute settings" on page 15.
- **You can disable the editor with a checkbox at the bottom** This will disable the editor for any future editing actions, whether you click OK or Cancel. The editor will remain disabled until you re-enable it through your general settings. For more information, see "General settings" on page 10.

Tip: In the attribute editor, you can double-click an existing value to edit it. A new popup editor will appear with a drop-down list with the same available choices as the attribute editor, as applicable for the respective attribute.

Custom modeless attribute editor

Structure tools includes a custom modeless attribute editor (**WS Structure Tools > Modeless Attribute Editor**), designed to replace the native modeless editor available from FrameMaker. It offers many improvements that make attribute editing easier and more efficient. You may find that this editor is the best choice for nearly all attribute editing that you have to do.

Tip: "Modeless" means that the editor can remain open while you do other work, unlike "modal" dialog boxes which must be dismissed before you can resume any other work.



The functionality of the editor should be reasonably intuitive, noting the following:

- **Values are set as soon as you move them to the "Current values" list** There is no need to click any buttons afterwards.
- **The editor operates on the currently-selected element or elements** If multiple elements are selected:
  - Only top-level siblings in the element selection are affected, no descendants
  - All lists show all possible values for all selected elements. For example, the "Current values" list displays a combination of all values set on the selected attribute for all selected elements.

  - If you move a value to the “Current values” list, that value only is set on each selected element, but only if the respective element has that attribute at all.
  - The **Set** button applies the values in the “Current values” list to all selected elements. Remember that this list reflects a combination of all selected elements and may not reflect the settings on any given element, at least until you click **Set**. Otherwise, this button is not used for general editor operation.

- **The “Available values” list shows EDD-provided choices and any additional choices that you have configured** You can also use the “(add...)” option to add additional values to the list. These values are remembered for future editing actions during the current session but do not affect your configured attribute settings. For more information, see “Attribute settings” on page 15.

The editor includes a variety of options which you can set with the **Options** button. Again, these options should be generally intuitive, allowing simple experimentation to provide the best explanation. Note the following:

- Any time attribute names are alphabetized, this applies to the editor only. Attribute order in the Structure View is controlled by the EDD and the editor never alters an EDD. For values, however, alphabetization applies both to the editor and the respective document.
- The “auto-collapse” option reduces the attribute display in the Structure View to populated and EDD-required attributes only.
- The “restore original available values” option causes the editor to remove any values that you added with the “(add...)” option in the “Available values” list, when the editor is closed. Otherwise, they are remembered for the duration of the FrameMaker session.
- The “preferred attributes” list is an editing convenience only, providing a means of placing certain attributes at the top of the “Attributes” list. It does not affect attribute order in your documents.

Copying and pasting attribute values

In certain right-click menus such as those which appear over the Structure View window, Structure Tools provides some commands for copying and pasting attribute values. These functions should be generally intuitive, noting the following:

- The clipboard used for copying data is independent of the normal Windows clipboard.
- When copying, all attributes and values for the selected element are copied. When pasting, all attributes and values are pasted, except for any attributes that the selected element(s) do not have.
- Technically, if an attribute on the special clipboard has no values, a paste action should delete any existing values on the target element, making its corresponding attribute empty as well. However, this will not occur unless you have your general settings configured to allow it. For more information, see “General settings” on page 10.
- The **Copy All Markup** command is identical to the **Copy Attributes** command, except that it also copies the element tag to the special clipboard. When pasted, all affected elements are changed to this tag.

- When pasting, you can paste to multiple selected siblings at the same time. If any of those siblings are text nodes, they are ignored, even if the special clipboard includes an element tag.
- The keyboard shorcuts for all these commands are configurable within the “additional settings” file (**WS Structure Tools > Open Additional Settings File**). See the contents of that file for more information.

Chapter 4 - External Calls to Structure Tools

Like many FrameMaker plugins, you can make external calls to Structure Tools to invoke certain plugin activities, often for purposes of automation. Specifically, you can call this plugin to:

- Launch the modal attribute editor (“LaunchModalAttrEditor” on page 27)
- Run the book processing function (“ProcessBook” on page 28)
- Set up parameters that affect other functions and behavior, such as parameters for a book processing run (“SetParm” on page 30)

These functions are fully exposed through the FrameMaker API and allow you to programmatically mimic the behavior of the plugin as used interactively through the GUI.

How to send an external call to Structure Tools

To call Structure Tools, you can use one of three methods:

- **With the FDK F_ApiCallClient() function, from another API client** If you are working on another FDK client, you can use `F_ApiCallClient()` to call Structure Tools. This function is part of the normal FDK library and does not require any changes to your normal project settings. For more information on the function itself, see the *FDK Developer’s Reference* provided by Adobe with the FDK.
- **With ExtendScript** Built-in with FrameMaker since FrameMaker 10, ExtendScript is a comprehensive scripting interface that can make calls to API cliends with the `CallClient` method. When called from ExtendScript, Structure Tools behaves identically to a regular API call.
- **With FrameScript** FrameScript®, a scripting tool by Finite Matters, Ltd®, has a comparable function for calling FDK clients, `CallClient`. When called from FrameScript, Structure Tools behaves identically to a regular API call.
- **With FrameAC** FrameAC by Mekon® (www.mekon.com) is a COM-based utility that enables developers to use Visual Basic to control FrameMaker. FrameAC also provides the ability to script calls to other API clients.

For any supported operation, you pass a string to Structure Tools which contains a command and any applicable parameters, and Structure Tools sends back a numeric code indicating the results. The syntax of these strings is the same for either API or scripting calls, and is explained in detail in this document.

Note: The call descriptions and examples in this document are written from an FDK/API perspective, using `F_ApiCallClient()`. If you are using FrameScript or FrameAC, the basic call syntax will be the same, sent using the mechanism supported by the respective tool.

General information on external calls

Before you attempt an external call, note the following:

- Certain commands require that you specify a document or book, which can be done by one of three methods. For more information, see “Specifying document and book arguments” on page 26.
- The default delimiter string between arguments in a call is three dashes (`---`). You can change the delimiter with `SetCallDelimiter`.
- Several calls to Structure Tools return zero (0) to indicate a command failure, consistent with the behavior of other FDK functions. However, F_ApiCallClient() also returns zero if it fails to communicate at all with the specified API client. If you aren’t sure whether your calls are reaching Structure Tools, you can call the special `Hello` command to verify that communications are getting through.
- With minor exceptions as noted in this document, call string arguments are not case-sensitive.
- To effectively use the external interface to Structure Tools, you should be familiar with the functionality and workflow of the plugin through the GUI.

Specifying document and book arguments

When a document or book identifier is required, you may use any of the following three methods:

- **An object handle ID** - The integer form of the F_ObjHandleT object ID for the file.
- **A filename** - A non-qualified filename, such as `MyDocument.fm`.
- **A file path** - A fully-qualified file path, such as:

  ```
  C:\MyDocs\MyDocument.fm
  ```

  With this method, you may substitute forward-slashes for backslashes. For example:

  ```
  C:/MyDocs/MyDocument.fm
  ```

In all cases, the file must be currently open. Structure Tools will not open any files.

Specifying Boolean arguments

When an argument requires a Boolean true or false, you can specify it as follows:

- For **true**, you can specify `1`, `true`, or any word that begins with “t”, including just `t`.
- For **false**, you can specify `0`, `false`, or any word that begins with “f” (yes, any word), including just `f`.

Boolean arguments are not case-sensitive.

Call reference

This section details the external calls you can make to Structure Tools.

Hello

Tests whether Structure Tools is initialized and receiving external calls.

Syntax

```
F_ApiCallClient("WS_StructureTools", "Hello");
```

Returns

Value	Meaning
0	Structure Tools is not initialized and/or communication failed. Possible causes include: • Structure Tools is not running at all. Check the FrameMaker interface for a WS Structure Tools menu. • You have used a plugin name that differs from the one registered in the `maker.ini` file. In the first argument of the `F_ApiCallClient()` function, you must specify the plugin name exactly as it is registered in `maker.ini`.
1	Structure Tools is ready.

LaunchModalAttrEditor

Launches the custom modal attribute editor for a specific element. Once launched, it must be manually dismissed.

Syntax

```
F_ApiCallClient("WS_StructureTools",
  "LaunchModalAttrEditor---Document---ElemId---DoMessaging");
```

where:

Document	(Optional) Document that contains the element for which the editor should be launched. If unspecified, the command attempts to use the active document, if one exists. For more information on specifying this parameter, see “Specifying document and book arguments” on page 26.
ElemId	(Optional) The F_ObjHandleT object ID of the element to edit. If unspecified, the command attempts to derive the element ID from the current insertion point location of the *Document*.
DoMessaging	(Optional, Boolean) Indicates whether to produce any interactive messaging in the case of errors during the launch of the editor. The default is true.

Returns

Value	Meaning
0	Communication with Structure Tools appears to have failed. Use Hello to test connectivity. This error can also occur if the launch failed for an unknown reason.
1	Editor launched successfully. Note that for synchronous processing, you will not receive this return until the editor is dismissed.
101	Unrecognized command. Check the syntax of the command itself.
103	Incorrect number of arguments sent with command.
104	Bad document argument. See “Specifying document and book arguments” on page 26.
105	Bad element ID argument.

ProcessBook

Launches the “book processing” function using any settings previously configured with SetParm.

Syntax

```
F_ApiCallClient("WS_StructureTools", "ProcessBook---Book");
```

where:

Book	(Optional) Book to process. If unspecified, the command attempts to process the currently active book, if one exists. For more information on how to specify this parameter, see “Specifying document and book arguments” on page 26.

Usage description

ProcessBook runs the book processing function start to finish, based on any options currently configured with SetParm. Before issuing ProcessBook the first time, it is highly recommended that you issue SetParm with the pb_restore_defaults option, then purposefully set the book processing options specific to your needs. Otherwise, you may not be able to predict which processing functions actually run and/or how they run, because any lingering settings from a GUI run and/or startup defaults may have an effect. For more information, see “Sample sequence for setting up a book processing run” on page 33.

For more information on how the book processing function works, see “Book processing with Structure Tools” on page 5.

Returns

Value	Meaning
0	Communication with Structure Tools appears to have failed. Use Hello to test connectivity.
1	Processing completed successfully. Note that this does not indicate whether any specific function (such as a book update) completed without error, only that the ProcessBook sequence completed without any unusual circumstances.
101	Unrecognized command. Check the syntax of the command itself.
103	Incorrect number of arguments sent with command.
104	Bad book argument. See “Specifying document and book arguments” on page 26.
113	One or more book components are currently closed. The process book function cannot run on a book unless all components are open, unless you set pb_open_closed_files=True with SetParm.

SetCallDelimiter

Changes the delimiter for external call arguments. The default upon startup is three dashes (“---”). Note that you can set a delimiter to whitespace, but afterwards all whitespace in a command string will be considered a delimiter. Therefore, do not use whitespace if your command arguments might contain whitespace.

Syntax

```
F_ApiCallClient("WS_StructureTools",
  "SetCallDelimiterNewDelimiter");
```

Note: The new delimiter directly follows the SetCallDelimiter command. Do not separate them with the old delimiter. Anything following the command will be considered the new delimiter.

Returns

F_ApiCallClient() returns one of the following values:

Value	Meaning
0	Communication with Structure Tools appears to have failed. Use Hello to test connectivity.
1	Delimiter was set successfully.
103	Command was sent without a new delimiter.

SetCallDelimiter syntax example

```
F_ApiCallClient("WS_StructureTools", "SetCallDelimiter++++");
```

SetParm

Sets a parameter related to upcoming activities planned for Structure Tools, such as a book processing run.

Syntax

```
F_ApiCallClient("WS_StructureTools", "SetParm---Parm---Value");
```

...where the following parameter/value pairs are supported for book processing, noting the following:

- For more information on setting Boolean arguments, see “Specifying Boolean arguments” on page 26.
- For book processing, it is recommended that you issue `pb_restore_defaults` first, then set up the desired processing configuration. For more information, see “Sample sequence for setting up a book processing run” on page 33.
- For additional details on book processing options, see “Book processing with Structure Tools” on page 5 and “Structuring (restructuring) generatable files” on page 8.

Parameter	Valid values and description
`pb_restore_defaults`	(No value required) - Resets all book processing options to their defaults, which effectively disables everything and clears any associated setting lists, such as show/hide condition lists.
`pb_open_closed_files`	Boolean - Indicates whether Structure Tools is allowed to open files that are currently closed.
`pb_close_opened_files`	Boolean - Indicates whether Structure Tools should save and close any files that it opened, after processing them.
`pb_hide_element_boundaries`	Boolean - Indicates whether to hide element boundaries.
`pb_refresh_element_definitions`	Boolean - Indicates whether to refresh element definitions and remove format overrides.
`pb_save_files`	Boolean - Indicates whether to save all files after processing.
`pb_hide_condition_indicators`	Boolean - Indicates whether to hide conditional text indicators.
`pb_show_hide_conditions`	Boolean - Indicates whether to show and/or hide conditions, based on the internal lists configured with `pb_add_show_condition` and `pb_add_hide_condition`.
`pb_add_show_condition`	Any condition name - Adds a condition name to show during processing. To clear the current list, issue `pb_clear_list`. Note that this setting has no effect unless you have turned on the show/hide function with `pb_show_hide_conditions`.

Parameter	Valid values and description
`pb_add_hide_condition`	Any condition name - Adds a condition name to hide during processing. To clear the current list, issue `pb_clear_list`. Note that this setting has no effect unless you have turned on the show/hide function with `pb_show_hide_conditions`.
`pb_do_book_update`	Boolean - Indicates whether to perform a book update. If set to false, all settings associated with a book update are ignored.
`pb_pre_framescript` `pb_post_framescript`	Any FrameScript name (case-sensitive) - Indicates FrameScripts to run before and after processing, respectively. ***Note:*** If you are calling Structure Tools from FrameScript, the behavior may be unpredictable if you set up additional scripts to run here. Therefore, these settings are recommended for Structure Tools calls from API clients only.
`pb_update_numbering`	Boolean - If `pb_do_book_update` is enabled, indicates whether to update numbering during the update.
`pb_update_text_insets`	Boolean - If `pb_do_book_update` is enabled, indicates whether to update text insets during the update.
`pb_update_xrefs`	Boolean - If `pb_do_book_update` is enabled, indicates whether to update cross-references during the update.
`pb_update_ole`	Boolean - If `pb_do_book_update` is enabled, indicates whether to update OLE objects during the update.
`pb_update_master_pages`	Boolean - If `pb_do_book_update` is enabled, indicates whether to update page layouts during the update.
`pb_regenerate_files`	Boolean - If `pb_do_book_update` is enabled, indicates whether to regenerate generatable files during the update.

Parameter	Valid values and description
`pb_regeneration_option`	If `pb_regenerate_files` is enabled, specifies how regeneration and restructuring should occur. Valid values are: • `skip_currently_structured` (default) - Regenerate all except files currently structured and leave regenerated files unstructured. • `regenerate_all_restructure_none` - Regenerate all files and leave all unstructured. • `regenerate_all_restructure_structured` - Regenerate all files but only restructure those that were structured originally. • `regenerate_all_restructure_all` - Regenerate all and restructure all.
`pb_conversion_table_path`	If the `pb_regeneration_option` indicates a restructuring action, the path to the conversion table to use for restructuring. Note that: • You can specify a full path or a filename. In the case of a filename, the plugin will look in the current book folder for that file, as with any book processing action (see "Structuring (restructuring) generatable files" on page 8). • If the path specified does not represent a fully qualified, valid filepath (such as a filename only), the plugin cannot know whether it will resolve upon book processing. Therefore, the external call will return 115 instead of 1. However, the path will still be set and the plugin will attempt to resolve it with each processing action. • The path can be specified with forward slashes or backslashes.

Additional options that may be set with this command include:

Parameter	Valid values and description
`modal_attr_editor_enabled`	Boolean - Indicates whether the custom modal attribute editor is enabled. If not, it will never appear for any element or attribute editing actions. This setting corresponds to the **Enable custom attribute editor** setting in the general preferences (see "General settings" on page 10).

Returns

Value	Meaning
`0`	Communication with Structure Tools appears to have failed. Use `Hello` to test connectivity.
`1`	Parameter was set successfully.
`101`	Unrecognized command. Check the syntax of the command itself.
`103`	Incorrect number of arguments sent with command.
`109`	Unrecognized parameter.
`110`	Illegal value for the specified parameter.
`115`	An invalid file path was specified for a call to set a conversion table path. **However, note that the path was still set, so the call can be considered successful. The plugin will attempt to resolve the path with each processing action.**

Sample sequence for setting up a book processing run

```
F_ApiCallClient("WS_StructureTools",
    "SetParm---pb_restore_defaults");
F_ApiCallClient("WS_StructureTools",
    "SetParm---pb_hide_element_boundaries---true");
F_ApiCallClient("WS_StructureTools",
    "SetParm---pb_show_hide_conditions---true");
F_ApiCallClient("WS_StructureTools",
    "SetParm---pb_add_hide_condition---ProductA");
F_ApiCallClient("WS_StructureTools",
    "SetParm---pb_add_hide_condition---ProductB");
F_ApiCallClient("WS_StructureTools",
    "SetParm---pb_do_book_update---true");
F_ApiCallClient("WS_StructureTools",
    "SetParm---pb_regenerate_files---true");
F_ApiCallClient("WS_StructureTools",
    "SetParm---pb_regeneration_option---regenerate_all_restructure_all");
```