Mario Ruggier
5 July 1999

Guide for using the Software Documentation Layout Templates

Version 2.0

This document has been prepared with Release 5.5 of the Adobe FrameMaker® Technical Publishing System using the User's Guide template prepared by Mario Ruggier of the Information and Programming Techniques Group at CERN. Only widely available fonts have been used, with the principal ones being:

Running text:	Palatino 10.5 pt on 13.5 pt line spacing
Chapter numbers and titles:	AvantGarde DemiBold 36 and 24 pt
Section headings	AvantGarde DemiBold 20 pt
Subsection and subsubsection headings:	Helvetica Bold 12 and 10 pt
Captions:	Helvetica 9 pt
Listings:	Courier Bold 9 pt

Preface

The problem

Documents are produced in all phases and activities of a software development cycle, e.g. User Requirements, Design and Architecture, Configuration Management, Test Plan, Project Plan, and so on. The content material required by each of these different document types is very specific – in fact a detailed table of contents, as well as additional guidelines, are normally provided by the standard being followed, e.g. IEEE, ESA, ISO. However, it is generally desirable to have a consistency in the layout (formatting, style sheets, visual style) across all of the specification documents for a software development project.

Separate layout and content templates

The Software Documentation Layout Templates (SDLT), produced in IT/IPT, are a set of generic format templates to be used in conjunction with separate content-only templates, specific to each document type. A content-only template will therefore specify such things as the detailed table of contents, sample material and guidelines for the type of document. The advantages offered by this scheme are:

– Visual consistency across the various documents for a project can be achieved in a manageable way, and requires the maintenance of only one set of layout templates.

– Allows for the independent production and maintenance of content-only templates – to add a template for a particular document type will require only the creation of a content-only template. This greatly reduces the effort needed both for adding as well as maintaining new templates.

– If a content-only template for a particular document type does not yet exist, 3rd parties have the liberty and the means to add it, as required by their project.

An example of such a content-only template is the new version of the ESA URD content template, which thus supersedes the last release of the CERN PSS-05 template, Version 2 of 30 June 1998.

Main features of the SDLT

The SDLT are produced for Adobe FrameMaker 5.5, thus usable on various UNIX, PCs and MACs. Features include:

– FrameMaker book structure.

– Pre-defined formats for cover pages, front matter, table of contents, automatically updated lists, chapters, appendices, and index.

– Comprehensive pre-defined formats for all typical software specification documentation material.

– Automatic generation of structured webs.

– Comprehensive user documentation.

Working model

The SDLT assume the following working model:

1. Identify the type of document that you need to produce for your project.
2. Choose the appropriate SDLT template, i.e. for longer documents use the book package, for short documents use the single file template.
3. If appropriate or available, choose the content template that will contain sample material and content-specific instructions.
4. Copy the layout templates to a clean directory.
5. Combine the chosen content template for your document with this copy of the layout templates (Section 1.5, "Combining with the content template").
6. Set up the book as desired: one chapter per file, or all chapters in the same file, etc. (Section 1.8, "Adding and deleting chapters").
7. Edit the document, following the content-specific instructions provided with the content template and the layout-specific instructions in this manual.

The different components of the above working model are illustrated in the figure below.

More information and availability

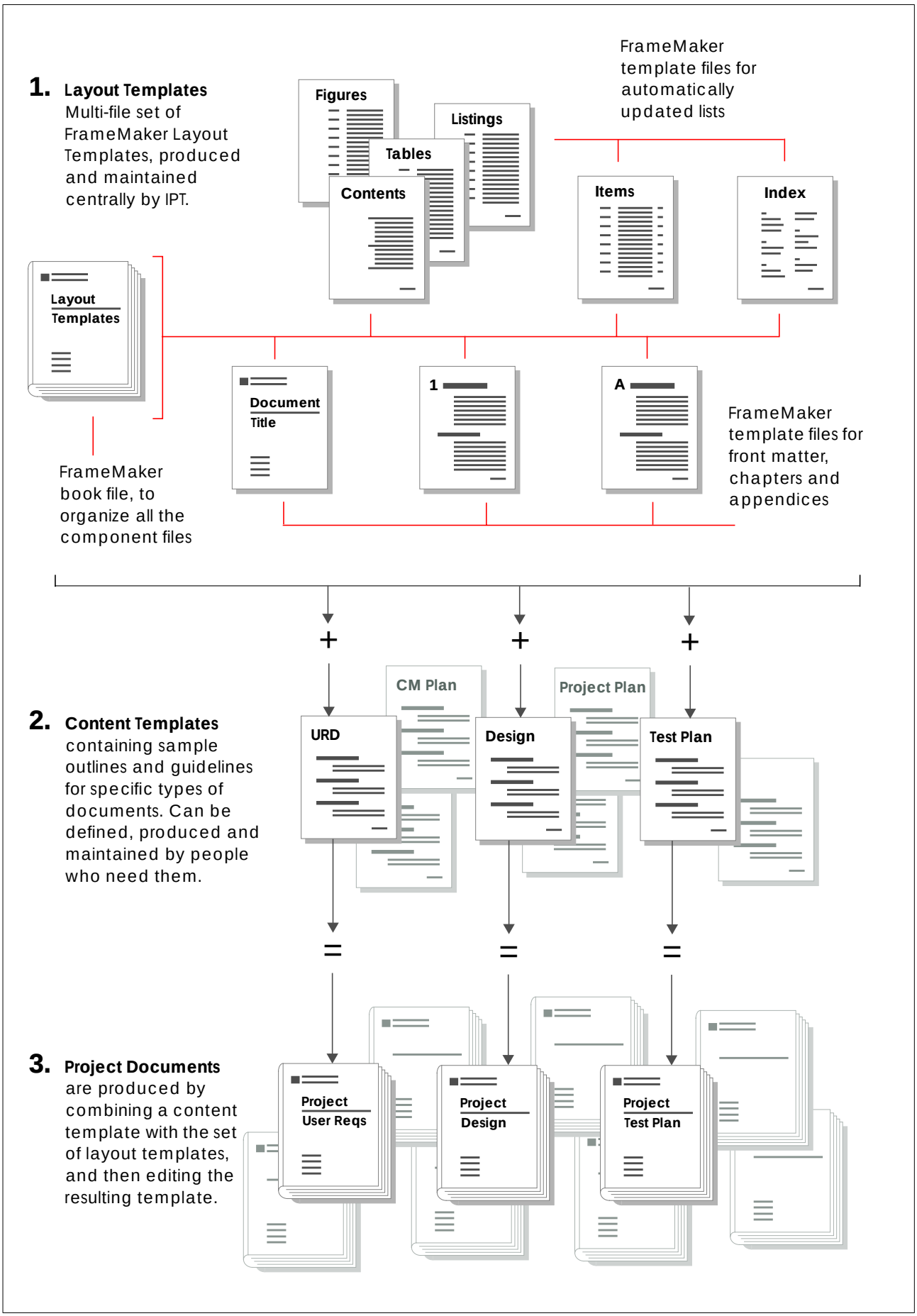
More information about the SDLT package, as well as access to distribution file, can be had from the web pages at:

`http://framemaker.cern.ch/sdlt/`

Feedback, problems and suggestions

The SDLT are supported by the IPT Group of the IT Division. Please send your problem reports, comments and suggestions to:

`DocSys.Support@cern.ch`



Specific project documents (3) are produced by editing templates obtained by combining the corresponding content template (2) with the software documentation layout templates (1).

About this manual

This manual provides the instructions needed to use SDLT, for both the book package as well as the single-file version*. It serves as the getting started guide, user's guide and as reference manual for the templates. Although it also contains several FrameMaker-specific tips, it is not a manual for using Adobe FrameMaker – for using FrameMaker please consult the FrameMaker documentation.

Intended audience

The SDLT, and so also this manual, have primarily two user types:

- The **editor**'s role is that of preparing a specific document for a project. It consists of:
  - gathering the required information;
  - organizing it into a cohesive document, according to the type of document, using the predefined formats of the SDLT;
  - managing the FrameMaker files, as well as any other supporting external files;
  - orchestrating the distribution of work and the interaction between the participating editors in the case that there is more than one.
- The **content template maker** defines and implements new content templates using the layout features of the SDLT, for other editors to use with the SDLT. The content template maker must understand how the generic predefined formats of the SDLT may be used to layout the particular type of content at hand.

Content template makers must also be editors, or at least be fully familiar with their role. In addition, content template makers must define formatting conventions for new type of material, using the predefined layout definitions.

Most of the information in this manual is directed at editors. Chapter 6, "Making content templates", provides the special extra information needed by content template makers.

* Instructions that are specific to the single-file version of the SDLT are included in the text of the single-file template itself.

Glossary

Book document A multi-file document, composed of a FrameMaker book file and of several component files, typically one for front matter, one for the table of contents, one or more for the chapters, one or more for the appendices, and one for the index, if any.

Book template One FrameMaker book file and the several component template files, one for each type of component.

Component file Single FrameMaker file that is part of a FrameMaker book. May be of several types, such as front matter, chapter, appendix, etc. A single component file may contain from one to several chapters (or appendices).

Component template Single FrameMaker file containing the formats for a particular type of component.

Content template A FrameMaker document containing sample content, and guidelines how to collect, organize and format the information, for a particular kind of software development document, such as a User Requirements Document, Architecture and Design, Test Plan, Project Plan, ...

Layout template One or more FrameMaker files that define the formats required for a particular type(s) of component file(s).

Project document A deliverable document, produced during the lifecycle of a software development project. May be a single FrameMaker file or a multi-file FrameMaker book.

Single file template A template file that includes all the formats necessary for short single-file documents, e.g. formats for the title page, for chapters and appendices, etc. Thus a single-file template is different from a component template as these are streamlined to contain only those formats for a specific part of a long document, such as only for a chapter or only for an appendix.

Contents

Chapter 1

Getting Started

This chapter introduces the editor to using the SDLT. We first define a step-wise procedure for getting started, after which we discuss all organizational and set-up issues.

1.1 Editor's road map

Follow the procedure below to get going. If you are not familiar with working with multi-file FrameMaker books, or templates, or with any of the actions required in any of the steps below, then refer to the indicated sections in this manual where related details may be found.

1. Familiarize yourself with the working model of the SDLT, illustrated in the section "Working model" in the preface of this document. In particular make sure you understand the associated figure.

2. Establish the type of document that you need to produce for your project, and get the approval of your project manager.

3. Decide whether to use the book version (for long documents with a global table of contents, index, etc.) or the single file (convenient for documents of few pages).

4. If appropriate, choose the content template. This will contain sample material and content-specific instructions.

   ⇒ Section 1.3, "Which content template?".

5. Get a copy of the layout templates, into a clean working directory. Give your working directory an appropriate name.

   ⇒ Appendix A.1, "The distribution file".
   ⇒ Section 1.4, "File organization".

6. Combine the chosen content template for your document with this copy of the layout templates. This is done by adding the content template file or files as components to your book document.

   ⇒ Section 1.5, "Combining with the content template".

7. Set up the book as desired. For example you may decide to have one chapter per file (if the chapters are big, or if each chapter has a different editor), or to put all the chapters in the same file. You may also change the name of the book file, as well as the component files.

   ⇒ Section 1.4, "File organization".
   ⇒ Section 1.8, "Adding and deleting chapters".
   ⇒ Chapter 4, ”Automatically updated lists”.

8. Edit the document, following the content-specific instructions provided with the content template and the layout-specific instructions in this manual.

   ⇒ Chapter 2, “Editing the front matter”.
   ⇒ Chapter 3, “Laying out the content”.

1.2 Which layout template?

The SDLT are available both as a FrameMaker book package, and as a single file template. Essentially both templates contain the same set of pre-defined formats, so it is easy to exchange material from one to the other (just copy and paste).

For short technical documents, that do not require a table of contents or an index, a single file template is more convenient to use than a multi-file book template package.

For longer documents, one should use the book package with the automatically updated lists of content, index, etc. This manual provides explanation mostly for using the book package. To use the single file template is the same, but easier. Extra specific instructions are available in the single file template itself.

1.3 Which content template?

Identify the document type that you need to produce. If a content template for this exact document type exists, use it. Otherwise look at other similar documents and define what material should go into your document, and how it should be organized.

Note that for general documents there are no content templates. In this case just use the SDLT as a generic template.

For more information on defining and producing new content templates, look at Chapter 6, “Making content templates”.

1.4 File organization

The way you organize your files depends on your process, i.e. on how you want to work. It is however highly recommended that all files for a document are placed in a separate directory, and that all imported files are separated into a subdirectory, e.g. **`imported`**.

The layout template files

The layout templates are organized as a FrameMaker book, with several component template files, some of which are templates for editable files (front matter, chapters and appendices) and the others being templates for automatically updated lists. You should have access to a reference copy of the template files at all times, as you may need to copy component templates while setting up your book, or when adding chapters.

See also Appendix A.1, "The distribution file".

Your document

For single authors a flat directory structure, similar to the structure of the template files, is the easiest. However for multiple authors it is best to have a subdirectory per chapter, as this is easier to manage access rights for, and to re-assemble the book or (in case of configuration management) to check-in and check-out individual chapters.

For single editor documents, organize your files as follows:

1. Copy either the file **`sdltbook.tar`** or the file **`sdltsinglefile.tar`** to the space where you will work.
2. Unpack this file. On UNIX:

   ```
   tar xvf sdltbook.tar
   tar xvf sdltsinglefile.tar
   ```

3. The command above creates a subdirectory such as **`sdltbook`**. Rename this subdirectory with a name appropriate for your document.
4. For books, rename the FrameMaker book file, **`sdlt.bk`** (that you will find in this subdirectory) to, for example, **`mydoc.bk`**. For single file documents, rename the file **`sdltsinglefile.fm`** to something like **`mydocument.fm`**.
5. Run FrameMaker and open the book file **`mydoc.bk`** or **`mydoc.fm`**.
6. Set up the chapters and appendices as explained in Section 1.8, "Adding and deleting chapters".
7. Remove, from the book file, any of the automatically updated lists if these are not needed. These may be re-added later. For more information see Chapter 4, ”Automatically updated lists”.

1.5 Combining with the content template

A content template contains:

a. sample material;

b. instructions to fill out the document;

c. in some cases, the specific content matter may also require predefined high-level formatting objects, e.g. a very specific kind of table that will be used over and over again.

Items (a) and (b) are incorporated by simply copying and pasting as explained below, but item (c) may require specific manual steps. Any such specific steps can be found in the little guide included with the content template.

To combine a content template with the layout template, do as follows:

1. Set-up your working location and FrameMaker files for your document, as indicated in Section 1.4, "File organization".
2. Make sure you have access to a copy of the files for the content template. If not, obtain a copy.
3. Open the chapter component of the content template, and:
   a. click anywhere in the text
   b. select **Edit / Select All in Flow**
   c. select **Edit / Copy**
   d. close the chapter component file
4. From the book file for your document, double click on the component file, that corresponds to numbered chapters, to open it, and:
   a. click anywhere in the text
   b. select **Edit / Select All in Flow**
   c. select **Edit / Paste**
   d. save the chapter component file for your document
5. Repeat steps 3. and 4. for the front matter and appendix components of the content template, if any.
6. Check the short guide included with the content template and follow any special steps required to incorporate formats specific to this content template.

1.6 Which style of headings?

By default, the headings for the book template are appropriate for long documents, with chapters being typically more than a few pages long. This style of headings may not be appropriate for documents of medium length, i.e. with chapters typically being only 1 or 2 pages long. In this second case you may prefer to use a smaller style for the headings, with chapters or appendices not necessarily starting on a new right page. A set of format files is provided for both long style and short style headings. To switch from one style of headings to another you just need to import the chosen style formats into your template.

Table 1.1 Two sets of heading format files, one for long and one for short style headings.

Format files for large style headings	Format files for small style headings

`lib/headings-big-autofiles.fm`
`lib/headings-big-chapter.fm`
`lib/headings-big-appendix.fm`
	`lib/headings-small-autofiles.fm`
	`lib/headings-small-chapter.fm`
	`lib/headings-small-appendix.fm`

To switch your template from the default large style of headings to the small style, import the short style headings into your component template, as follows:

1. Open the book file of the template.
2. Shift-click on the book **File** menu and select **Open All Files in Book**.
3. Open the file **`lib/headings-small-autofiles.fm`**.
4. From the book **File** menu select **Import / Formats...**
5. For **Import From Document:** select **`headings-small-autofiles.fm`**.
6. Under **Import and Update:** *de*select everything except for **Paragraph Formats**.
7. Under **While Updating, Remove**: select **Manual Page Breaks** and **Other Format/Layout Overrides**
8. Under **Update:** remove all files (double-click) that are not automatically updated, i.e. under this column leave only the files:

   `sdltTOC.fm`
   `sdltLOF.fm`
   `sdltLOT.fm`
   `sdltLOL.fm`
   `sdltLOI.fm`
9. Click on **Import.**
10. Repeat steps 3. through 9., for the chapter and appendix component templates (thus replacing filenames as appropriate).

Note that if paragraph formats are updated from a new version of the template, this procedure will need to be repeated.

1.7 No template customizations!

Template customizations are formatting modifications involving the creation of new formats or of format overrides*. Thus, creating a new paragraph tag is a template customization, and changing the formatting of a single paragraph, such that it does not correspond anymore to the corresponding definition in the paragraph catalog, is an example of a format override.

A mark-up and formatting standard is absolutely required if different people are to maintain project documents. Adhering to the standard implies (amongst other things) that only tag names and formats defined in the standard template are used. Format customizations create problems such as:

- Confusing inconsistencies between this document instance and the standard template and its documentation. Other editors (as well as yourself!) will be utterly confused next time the document is maintained...
- Format catalogs, such as for paragraphs and characters, get cluttered with possibly badly chosen and undocumented names, reducing usability.
- Any improvements or corrections to the standard template cannot be propagated properly (by using import formats) to documents with overrides.
- When importing formats from an updated version of the standard template, overrides (customizations of formats that retain the standard tag names) will be all lost.
- Customizations may require other unexpected changes to the template, such as:
  - the automatically updated lists, e.g. TOC, list of figures;
  - automatic running headers;
  - HTML conversion rules on reference pages, i.e. conversion to WWW will break.

In other words, **no template customizations!**

1.8 Adding and deleting chapters

Adding a chapter is as trivial as adding a heading paragraph of level 1, i.e. adding an **H1NoNum**, **H1** or **H1App**, depending on whether the component file is of type front matter, chapter or appendix, respectively.

If you prefer to having one chapter per file, than adding a chapter also involves adding a file to the book, explained below. Adding / deleting appendices is exactly the same.

* For a more precise list of what is and what is not a template customization, please see Appendix A.2, "Acceptable additions of format tags".

Adding a new chapter file to the book

1. If you want the chapter to have its own subdirectory, then create one at the same location as the FrameMaker book file.
2. Copy the chapter template file from a reference copy of the layout templates to where you want the new chapter file to be, assigning it an appropriate name.
3. If the chapter is in its own subdirectory, create an **`imported`** subdirectory to hold any imported files for the chapter.
4. With FrameMaker open the book file for your document.
5. Select **File:Add File...** and browse to the newly copied chapter file.
6. Select the file and set the position in the book where it is to be added.
7. Click on **Add**.
8. Repeat 1. through 7. for each chapter to be added.
9. Update the running headers and footer, and any other variables, for the added files as detailed in "Setting the running headers / footers" of Chapter 5.
10. Update the book as explained in "Updating the FrameMaker book" of Chapter 5.

Restarting page numbering for first chapter file in the book

The page numbering of the first chapter file should be restarted at 1. This information is stored in the book file as part of the component file settings, and is set by doing:

1. In the book file window, click once on the filename for the first chapter file.
2. From the book file menu choose **File:Set Up File...**
3. Under **Page Numbering** choose **Restart at 1**.
4. Click on **Set**.

All other chapters should have **Page Numbering: Continue**, **Paragraph Numbering: Continue** and **Starting Page Side: Right**.

Deleting a chapter file from the book

1. From the book file select **File:Rearrange File...**
2. In the **Rearrange Files** dialog, click once on the file name you want to delete to select it.
3. Click on **Delete**. (This only removes the book reference to this file. If you want to remove it from disk you will need to do so separately.)

1.9 Managing imported files

FrameMaker allows you to import both graphic and textual files by copy or by reference. Importing by copy stores a description of the graphic or text inside the FrameMaker file, while importing by reference stores only the file system path information for the external file. Some arguments for and against importing by reference are presented in Table 1.2. Importing by reference is highly recommended.

Another recommendation is to keep the number of different graphic formats used to a minimum, ideally one or two for bitmap (e.g. JPEG for high quality photos and GIF for other lower quality bitmaps) and one for vector (e.g. EPS). Many different formats may be inconvenient when you need to process the images or move the document across platforms.

Table 1.2 A small comparison of the advantages for importing graphics by copy or by reference.

Advantages of importing by reference	Advantages of importing by copy
• The FrameMaker document is automatically updated when the graphic or text file changes. • Less duplication, e.g. if the same image is imported more than once in the same chapter it is not duplicated. • Keeps the size of the FrameMaker file itself small. • A list of all imported graphics may be generated: from book **FIle** menu, select **Add File...**, then generate a list of references using **Imported Graphics** as the source reference type.	• Guarantees that the FrameMaker file and the graphic stay together, thus the file is easier to move around than a directory structure with relative pathnames. • Some graphic file formats are not compatible across platforms. When a graphic is imported by copy it is stored internally as a FrameImage or FrameVector formats which can be displayed on all platforms.

Chapter 2

Editing the front matter

The front matter consists of the cover and inside cove, as well as the document preliminaries, such as abstract, document history, and preface. A separate template component file, **`frontmatter.fm`**, is provided for the front matter.

2.1 Cover page

The cover page, apart from its importance to the usefulness of the document, also specifies some of the document meta-data, such as the title, the version of the document, the authors, the document context, and so on. These units of meta-data are logically tagged with dedicated paragraph and variable formats. Thus, the cover page may contain only a sequence of logically-named paragraphs, each of which contains a similarly named user-defined variable, as indicated in Table 2.1.

Table 2.1 The complete ordered sequence of all paragraph formats that make up the book cover page.

Paragraph tag	Contains	
DocContextInfo	Variable, with extra context information, such as the group or collaboration in which this work is being done.	Optional
DocContext	Variable, with context information such as the general project (for which this project is a sub-project), or the group in which this work is being done.	Recommended
UtilClearRestOfColumn	Empty, used only to force next paragraph to start at top of next text column.	**Required**
DocProjectName	Variable, with the name of this project. If this project is a subproject of a bigger project, then this should be the name of the subproject. Used as running header throughout the document.	**Required**
DocProjectInfo	Variable, with project sub-title, or extra information about the project.	Optional

Table 2.1 The complete ordered sequence of all paragraph formats that make up the book cover page.

Paragraph tag	Contains	
DocName	Variable, with things like Specification, User Requirements, Project Plan, etc.	**Required**
DocNameInfo	Variable, with extra information about this document.	Optional
DocVersion	Variable. Value must be incremented for all publications with changes more substantial than grammatical or spelling corrections.	**Required**
DocIssue	Variable. For a version, each issue contains only grammatical or spelling corrections.	**Required**
DocEdition	Variable, with language (English | French) and whether b&w or color.	Optional
DocStatus	Variable: Draft or Final or ...	Recommended
DocID	Variable, with string to identify the document, independent of the version and issue.	Recommended
DocDate	Variable, with date of last modification	Recommended
DocCreationDate	Variable, with date of document creation	Optional
DocAuthor	Variable, with the list of authors, or editors, or contributors... Type the appropriate text, such as "Authors:", followed by a tab, followed by a comma separated list of the authors. Forced line breaks may be used if one name per line is preferred.	Optional
PlaceHolder	Anchored frame, to contain a picture.	Optional
DocInstName	Variable, with the full name for CERN, or other institution.	**Required**

The **Document Control Sheet** (see Section 2.2, "Other front matter") organizes the complete document meta-data, as a reference for the reader of your document. Therefore, what meta-data you show on the cover page is up to you. You can simply insert / delete the necessary paragraphs and variables.

Editing the document meta-data

For each piece of meta-data, there is a pre-defined user variable. The easiest way to edit the document meta-data is to edit the definition of the corresponding user variables, listed on the **DocMetaData** reference page, in the **`frontmatter.fm`** component template file.

Propagating document meta-data to entire book

To propagate the modified user variable definitions, import the variables, and **only** the variables, from your **`frontmatter.fm`** file to all the other files in the book:

4. From the book **File** menu select **File / Import Formats...**
5. Under **Import from Document:** select (the open) `frontmatter.fm`.
6. Under **Import and Update:** deselect everything except for **Variables**.
7. Make sure that all component files in the book are listed under **Update:**.
8. Click on **Import**.

The document title

The document title is logically composed of the concatenation of the two paragraphs (and corresponding user-defined variables with the same names), **DocProjectName** and **DocName**. The text of these two variables are used for the running header on each page of the document.

To change the project logo

Go to the **Front Page** master page and remove the current project logo (but not the frame). Then either copy one of the project logos included in the reference file `lib/logos.fm` (included in the distribution), or just import one into the empty logo frame on the **Front Page** master page. Scale as needed. Finally, using **Graphics / Align**, align along **Tops** and **Right Sides**.

If, for some reason, you wish to have two logos here, then you can replicate the frame with the first logo onto the right side of the master page, appropriately aligned. However if you do this please remember to modify the user variables, **WWWInstitutionName1** and **WWWInstitutionName12**, that control how the logos will appear in the generated WWW version of the document (see Section 5.2, "Converting to HTML").

2.2 Other front matter

The frontmatter template includes a sequence of tables that provide information about the document. You may choose to either keep these sections or to delete them entirely. If you delete them and would later like to bring them, you may copy and paste them again from the **FrontMatterTables** reference page.

Abstract

The heading should be tagged with **AbstractH1** and the text with **AbstractBody**.

Control Sheet

The Document Control Sheet collects all the document meta-data in one table. All the information is stored in user-defined variables. Changing the value requires

changing the variable definition, as explained in the section above, “Editing the document meta-data”.

Status Sheet

The Document Status Sheet provides a history of versions and issues of the document. Add a table row for each version or issue. Note that the document title and the document ID should never change from one version / issue to another.

Change Record

The Document Change Record (DCR) lists the changes between this version and the previous version of a document. Use a table row for each change, and a new DCR per version. For each new added row, you will need to manually straddle the columns under "Reason for Change".

Note that changes between issues are not recorded, as these are only corrections of grammatical or spelling errors.

Distribution List

The Document Distribution List keeps a record of who has been given copies of a version of this document.

Chapter 3

Laying out the content

This chapter describes the general usage of the predefined FrameMaker formats in the template files. For how to use these formats to layout content-specific material, please consult the guidelines included with the content template you are using.

3.1 Headings

There are six levels of headings. However, it is not recommended to have documents organized in a 6-level hierarchy. As a rule of thumb, not more than four levels should be used. The complete list of headings are listed in Table 3.1

Table 3.1 The six levels of headings for the front matter template (not numbered), the chapter template (numbered numerically) and the appendix template (numbered alphabetically).

In front matter template	In chapter template	In appendix template
H1NoNum	H1	H1App
H2NoNum	H2	H2App
H3NoNum	H3	H3App
H4NoNum	H4	H4App
H5NoNum	H5	H5App
H6NoNum	H6	H6App

Sections of items may use another set of special headings (see Section 3.13, "Items" for more details).

Automatically updated lists also have special headings reserved for their use (see Chapter 4, “Automatically updated lists”).

3.2 Running text paragraphs

General running text paragraph tags are prefixed with **Body**. Table 3.2 lists all general text body paragraphs.

Table 3.2 The general running text paragraph formats.

Paragraph tag name	Description
Body	The basic text paragraph style.
BodyComment	Similar to the basic paragraph but the text font is italic.
BodyInset	Similar to basic text paragraph except that this one is indented both from the left and the right.
BodySmall	For information of secondary importance. Similar to normal text paragraphs except that the text font size is smaller than normal body text.
BodyVerbatim	For pieces of code or text that requires a fixed font. To be used in conjunction with hardspaces and forced line breaks.

3.3 Character highlights

Table 3.3 The predefined character highlights.

Character tag name	Description
Acronym	For stylish formatting of acronyms. Instead of normal uppercase, type lowercase letters – they will be formatted with slightly spaced small caps.
Bold	Forces bold weight. All other properties are left as is.
Changed	For manual revision control. Overlines text.
Code	For words requiring a fixed with font. Uses Courier.
DfnTerm	For highlighting definition terms. Text is shown in Helvetica 9.5 pt Bold, exactly as **DfnPart1Term** paragraph styles.
Emphasis	Emphasizes text by showing it in italic. Same as **Italic** tag but uses a logical name.
FigCallout	For graphic ascii lines, that need to look like text of **FigCallout** paragraphs.
FigLabel	For graphic ascii lines used as labels in figures.
Italic	Forces italic font style. All other properties are left as is.
Keyword	To highlight special words. Forces Palatino Bold Italic, with all other properties left as is.

Table 3.3 The predefined character highlights.

Character tag name	Description
NoSpellCheck	Language is forced to **None**, so that the text is not spell-checked. Appearance does not change.
Regular	Forces regular font weight, angle and variation. Used by the automatically generated lists.
Removed	For manual revision control. Strikes-through text.
Roman	Forces roman (regular) angle variation, for use in equations to manually reformat non-variables back to roman (see “Fine-tuning equation formatting” on page 22).
Strong	Highlights text by showing it in bold. Same as **Bold** tag but uses a logical name.
Subscript	Text is subscripted.
Superscript	Text is superscripted.
SymbolFont	Forces the Symbol font face. All other properties are left as is.
ToolOption	To markup tool menu choices, options, etc.
ZapfDingbatsFont	Forces the ZapfDingbats font face. All other properties are left as is.

3.4 Cross-references

Table 3.4 The list of pre-defined cross-references formats and indication of their use.

List paragraph tag	Description
Appendix paranum	To refer to sections in the appendix by their section number.
Appendix paranum, "paratext"	To refer to sections in the appendix by their section number followed by the paragraph text, in quotes.
Filename	Displays the filename, in italics, of the FrameMaker file containing the referenced paragraph.
ItemIdentifier	To refer to an item by its identifier.
Page	Display the page number of the referenced paragraph.
Paranum	To refer to any paragraph, by its autonumber string.
ParanumOnly	To refer to any paragraph, by the numeric value of its autonumber string (added text in the autonumber string is omitted).

Table 3.4 The list of pre-defined cross-references formats and indication of their use.

List paragraph tag	Description
ParanumParatext	Concatenates the autonumber and text of the referenced paragraph.
Paratag	Displays the tag name of the paragraph being referenced.
Paratext	Displays the text of the paragraph being referenced.
RefLiterature	To refer to one of the cited publications.
Section paranum	To refer to sections by their section number.
Section paranum, "paratext"	To refer to sections by their section number followed by the paragraph text, in quotes.

3.5 Bulleted and numbered lists

Bulleted, dashed and numbered list paragraphs, as well as two kinds of list continuation paragraphs, are available at 3 levels (Table 3.5). Alphabetic and roman lists are only available at the first level (Table 3.6).

List item types should be the same at any one level, but may be different from one level to another. However, lower level list items should always be contained in a list item that is one level higher.

Table 3.5 Three levels of bulleted, dashed and numbered lists, with two kinds of continuation paragraphs.

List paragraph tag	Description
LBullet1 LBullet2 LBullet3	Three levels of bullet item paragraphs.
LDash1 LDash2 LDash3	Three levels of dash item paragraphs.
LNumber1Fst LNumber1Nth LNumber2Fst LNumber2Nth LNumber3Fst LNumber3Nth	Three levels of numbered paragraphs. The first item in a numbered list uses a **...Fst** paragraph, to reset the automatic counter to 1.
LCont1 LCont2 LCont3	Three levels of list item continuation paragraphs.
LContVerbatim1 LContVerbatim2 LContVerbatim3	Three levels of list item continuation paragraphs for code excerpts or other material that requires a fixed width font.

Table 3.6 One level of alphabetic and roman list item paragraphs.

List paragraph tag	Description
LAlpha1Fst LAlpha1Nth	One level of alphabetic list items.
LRoman1Fst LRomanNth	One level of roman list items.

3.6 Definition lists

Information that has the form of term and description can be formatted using a term paragraph followed by a description paragraph. There are two term (Table 3.7) and three description (Table 3.8) formats, which may be combined together to get the required effect. Definition list paragraphs are available only for the first level.

A definition list item is formatted by using one term paragraph (of two styles), followed by one description paragraph (of three styles).

Table 3.7 Two kinds of term paragraphs, used in conjunction with the three kinds of description paragraphs.

Term paragraph	Description
Dfn1Part1Term	Forces the description paragraph that will follow to start on a new line.
Dfn1Part1TermRIH	Term paragraph behaves like a run-in head (RIH), i.e. allows the description paragraph that will follow to start on the same line.

Table 3.8 Three kinds of description paragraphs, used in conjunction with the two kinds of term paragraphs.

Term paragraph	Description
Dfn1Part2Desc	Left indentation of description is medium (2cm).
Dfn1Part2Desc0Indent	No left indentation of description (flush left).
Dfn1Part2Desc2Indent	Left indentation of description is high (4 cm).

Defining and using acronyms

Acronym definition lists are formatted using one of the above pairs of term-description paragraph formats, namely **Dfn1Part1TermRIH** followed by **Dfn1Part2Desc**, as indicated in Figure 3.1.

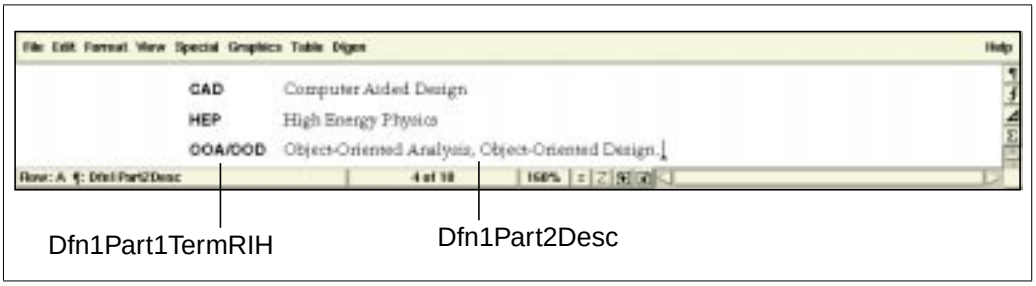


Figure 3.1 An example of definition list usage; formatting a list of acronyms.

When using acronyms in the text, you may improve the printed results by typing the acronym in lowercase and then tagging it with the **Acronym** character tag. The following two examples show the results obtained using the **Acronym** character tag (a) and just using standard uppercase (b).

a. ...by importing from CAD systems which conform to the STEP standard...

b. ...by importing from CAD systems which conform to the STEP standard...

3.7 Figures

We differentiate between two kinds of figures: normal figures (that occupy less than one full page) and large figures (that occupy a full page). The main reason for this distinction is that full-page figures must be floated*, while smaller figures are preferably not†.

Normal figures (half page or less)

Normal figures are formatted by first creating a **PlaceHolder** paragraph and then inserting a table using the **Figure** table format. An anchored frame is then inserted into the upper cell of the table, to contain the graphics for the figure.

The predefined table width of 15.3 cm should never be modified – the figure box should always be as wide as the running text column, i.e. 15.3 cm.

Large figures (full page)

A full-page figure is inserted by clicking in a text paragraph‡ and inserting a table using the **FigFullPage** table format. In the single cell of this table an **Anchored Frame**, with the exact properties that are specified in Table 3.9, is inserted. This

* In the case when it does not fit in the current column, then it is automatically moved to the next column that can hold it, while filling the remaining space of the current column with text that, in the document flow order, follows the figure.

† The most reader-friendly location for small figures is next to the text which refers to them. Full-page figures, by definition cannot be next to the text that refers to them.

‡ Full-page figures must **not** be anchored in an empty **PlaceHolder** paragraph.

frame forces the full-page figure, which is set to float, to occupy exactly the next full page.

Table 3.9 The exact properties of an anchored frame for a full-page figure.

Anchored Frame property	Setting
Anchoring position:	Below Current Line
Alignment:	Centered
Cropped:	On
Floating:	Off
Width:	17 cm
Height:	21.4 cm

Cross-referencing figures

Insert a cross-reference to a **FigCaption** paragraph, using the **Paranum** cross-reference format.

3.8 Tables

Tables, like normal figures and code listings, are always anchored in an empty **PlaceHolder** paragraph. A simple table format, **BasicTable**, is provided as convenient starting point. A set of predefined **Tbl...** paragraph formats are specially intended for use in table cells.

Customizing

Properties like straddling and irregular ruling may not be stored as part of a FrameMaker table catalog definition. You will therefore quite certainly need to customize tables, as dictated by your table content.

The predefined width of 15.3 cm should not be modified – tables should always be as wide as the running text column, i.e. 15.3 cm. Never make it narrower. If you must, you may make it wider – in this case make it as wide as the text frame for the page, i.e. 17 cm.

When customizing tables, you should keep the general look and feel consistent throughout the document.

Adding formats

Feel free to define new formats, with new tags, and add them to the catalog. This does not constitute a format override, as explained in Appendix A.2, "Acceptable additions of format tags".

Cross-referencing tables

Insert a cross-reference to a **TblCaption** paragraph, using the **Paranum** cross-reference format.

3.9 Code listings

We can group listings into the three categories shown in Table 3.10. Code is formatted as one line per paragraph. In the case of code line numbers, the **CodeLineFst** and **CodeLineNth** paragraph tags are used. Otherwise the **CodeVerbatim*** tag is used.

Table 3.10 The three categories of code listings and how to create them.

Category	Description	How to create	Defaults
Snippet	Composed of only two or three lines of code.	A **Snippet** table in a **PlaceHolder** paragraph	No caption. No code line numbers. Enclosed in a box, to be set off from main text.
Normal	About half a page long or less.	A **CodeListing** table in a **PlaceHolder** paragraph	Autonumbered caption. Code line numbers. Enclosed in a box, to be set off from main text.
Long	A full page or longer.	A **CodeCaption** paragraph, followed by a **CodeLine** paragraph per line of code.	Autonumbered caption. Code line numbers. Not enclosed in a box.

Controlling indentation and spacing

By default, the template files have **Smart Spaces**† turned on. This tells FrameMaker to not allow you to insert more than two or more adjacent spaces. In general this is useful. However for code listings you will generally want multiple spaces – to obtain multiple spaces use **HardSpaces**‡.

Importing a listing by copying

When you copy a piece of code into Frame, the original white space will be retained for you. However, when you then go and edit a line, any adjacent white spaces will be collapsed to a single space! To avoid this, replace all white space as follows:

---

* For proper conversion of **CodeVerbatim** paragraphs to WWW, line breaks within the same logical paragraph must be forced with hard returns (**Shift Return**) and **not** with normal carraige returns.

† **Format:Document:Text Options...**

‡ **`Control + Spacebar`**

1. Select the entire code listing
2. Pop-up the **Edit:Find/Change...**
3. Set **Find** to **Text:** and type a single space in the text field.
4. Set **Change** to **To Text:** and type a single hardspace in the text field.
5. Make sure that **Change All In:** is set to **Selection**.
6. Hit **Change All In:**
7. Accept the **Cannot be undone. OK to continue?** pop-up.

Importing a listing by reference

When importing text files by reference, any white spaces in the source are retained. The only adjustment you are allowed to do is to assign an appropriate paragraph tag to the entire inset, e.g. **CodeLineFst** or **CodeVerbatim**.

Tabular characters in the source file are problematic as tabs in Frame have a different meaning than tabs in ascii files. It is better to replace any tabs in the source file with the appropriate spaces. If this is not possible, then I suggest you create a special paragraph format to simulate the same pattern of tab stops in the source.

Cross-referencing listings

Insert a cross-reference to a **CodeCaption** paragraph, using the **Paranum** cross-reference format. Only code listings with a caption can be cross-referenced, i.e. **Snippets** cannot be referenced.

3.10 Mathematical formulae

All mathematical formulae should be formatted as real FrameMaker equations – do not imitate using strings highlighted with symbol font face, superscript and subscript, as this will introduce visual inconsistencies.

The FrameMaker equation editor

The FrameMaker equation editor is a structured mathematics editor with a graphic interface. Thus, a function must have operands, the square root symbol is not simply a symbol but an operator with an operand, and so on. For this reason it may be a little slow to get used to, especially if one is accustomed to typing LaTeX equations. However, all standard LaTeX commands for mathematical symbols are provided as FrameMaker shortcuts.

Small, medium or large equations

FrameMaker equations may be assigned a size of small, medium or large, which only affects their appearance. This allows the differentiation between stand-alone

equations, equations in-lined in the text, and equations in footnotes, as shown in Table 3.11.

Table 3.11 Usage of large, medium or small equations.

Equation size	Shortcut to create	Usage
Large	**Esc m l**	Numbered equations that stand alone. Use an **Equation** paragraph. To center, insert a tab character at the beginning of the line.
Medium	**Esc m m**	Equations inlined in the running text (normally quite small).
Small	**Esc m s**	Equations in footnotes or otherwise small text.

Cross-referencing equations

Insert a cross-reference to a **Equation** paragraph, using the **Paranum** cross-reference format. Only numbered equations can be cross-referenced, i.e. those contained in an autonumbered **Equation** paragraph.

Fine-tuning equation formatting

In general, equations in variables should be in italic, while subscripts should not. Subscripts that are themselves variables should be italic. However, all greek symbols should never be in italic irrespective of whether they are variables or subscripts. This convention is largely due to a lack of a universally available italic symbol font.

In this template, whatever FrameMaker thinks is a variable is formatted in italic. In some cases manual adjustment is necessary to reformat non-variables back to roman, using the predefined **Roman** character tag.

3.11 Bibliographical references

Bibliographical references are collected together in a reference section, as recommended by the content template you are using, and tagged with the **RefLiterature** paragraph tag. Use the appropriate formatting conventions, according to the type of document being cited. An example is indicated in Figure 3.2.

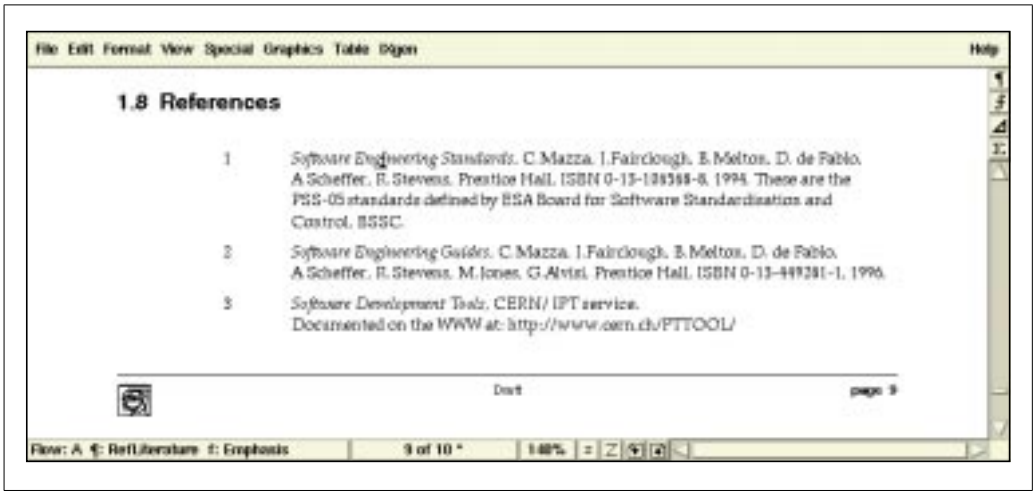


Figure 3.2 Example of a list of publication references.

To refer to a cited publication in the text of the document, insert a cross-reference to the citation (**RefLiterature** paragraph) using the **RefLiterature** cross-reference format. References to publications appear in the text with a number between square brackets.

3.12 Footnotes

Footnotes are inserted simply by clicking in the text and selecting **Special/Footnote**. A **Footnote** paragraph is automatically created either at the end of the page or, in case of tables, below the current table.

3.13 Items

The template contains a module of predefined formats for formatting generic objects that require all or any of:

a. an identifier;

b. a title;

c. a statement;

d. an explanatory block containing text and figures;

e. and a variety of attributes.

Formatting items

Table 3.12 The elements, in a logical order, that make up an item, and how to format them. Depending on the type of item you wish to format, you can choose to use and discard and of these elements.

Item Element	Description	How to format
Identifier	To specify a string that will identify the item.	**ItemIdentifier**
Title	A short title.	**ItemTitle**
Statement	A concise and complete statement of the item.	**ItemStatement**
Explanatory block	A section with supporting material.	Any and all paragraphs for running text, lists, figures and tables, etc.
Attributes	To specify attribute values for the item, for all items in a consistent way.	For each attribute, use a **ItemAttribName** followed by an **ItemAttribBody**.

Special heading for grouping items

A predefined template for a list of items is also provided. You can choose to include it or to remove it from your FrameMaker book. For more information see Chapter 4, “Automatically updated lists”.

This automatically updated list of items is a simple flat list. In the case of many items in a document, this automatically updated flat list of items may be made more useful if headings are used to break the long list into smaller sections. This can be done by selectively using the predefined special item headings in chapters and appendices, as opposed to normal headings. These headings are visually identical to normal headings and, in addition to showing up in the table of contents as expected, they also show up in the automatically updated list of items.

Table 3.13 Special item headings are available for sectioning the automatically updated list of items.

Normal chapter and appendix headings		Chapter and appendix item headings	
H1	H1App	ItemH1	ItemH1App
H2	H2App	ItemH2	ItemH2App
H3	H3App	ItemH3	ItemH3App
H4	H4App	ItemH4	ItemH4App
H5	H5App	ItemH5	ItemH5App
H6	H6App	ItemH6	ItemH6App

3.14 Other utility paragraph formats

Table 3.14 A few miscellaneous utility paragraphs and how to use them.

Paragraph tag	Usage
UtilAlertBeginNote	Makes a "note" icon appear in the sidehead area. Marks the beginning of a small alert section for a note. Should be followed by a label (e.g. Note) tagged with a **UtilAlertLabel**, one or more normal text paragraphs for the note text, and closed with an **UtilAlertTerminate** paragraph.
UtilAlertBeginWarning	Makes a "warning" icon appear in the sidehead area. Marks the beginning of a small warning alert section. Should be followed by a label (e.g. Warning) tagged with a **UtilAlertLabel**, one or more normal text paragraphs for the warning text, and closed with an **UtilAlertTerminate** paragraph.
UtilAlertLabel	To tag labels (or titles) of small alert sections.
UtilAlertTerminate	Draws a line to mark the end of a small alert section.
UtilClearRestOfColumn	Forces whatever comes after to start at the top of next text column. Useful when a page break is desired at special places, and the property is not built into the format that follows. Avoids page break format overrides.
UtilSideSticky	Use as a little yellow sticky in the sidehead area of the text column, for little notes.
UtilTDB	To record in the running text things that are still to be done or discussed.
UtilTemplateCredits	Only for single file template. A paragraph that should be retained at the very end of the document, and that makes a template credit sentence appear in a box.

Chapter 4

Automatically updated lists

The content of these files should never be edited manually. This chapter provides the information required for the maintenance of these files, namely what is required for these files to be re-added to the book. Once part of the book, all that is required for these files to be automatically updated is to **Generate/Update** the book, as explained in Section 5.1, "Routine book procedures".

The main heading for each of these auto-updated should use one of four heading tags reserved for this purpose, as indicated for each type in Table 4.1. The purpose for the different headings is to have the flexibility of placing an automatically updated list anywhere in the document – in the front matter as a non-numbered chapter, as a numbered chapter or as an appendix.

Automatically updated list files may be deleted from the book just like any other file (book **File** menu / **Rearrange Files...**). However, re-adding any of these lists (book **File** menu / **Add File...**) requires some list-specific information. Table 4.1 specifies exactly this information.

Table 4.1 The set-up data needed when re-adding an automatically updated lists to the FrameMaker book.

Type of list	To add	Sources	Filename Suffix	Starting Page Side	Main heading tag
Table of Contents	Book File / Add File... / Generated List: Table of Contents	AbstractH1 AutoH1App AutoH1Chap AutoH1NoNum H1, H1App, H1NoNum H2, H2App, H2NoNum H3, H3App, H3NoNum H4, H4App, H4NoNum H5, H5App, H5NoNum H6, H6App, H6NoNum ItemH1, ItemH1App ItemH2, ItemH2App ItemH3, ItemH3App ItemH4, ItemH4App ItemH5, ItemH5App ItemH6, ItemH6App	TOC	Right	AutoH1Contents

Table 4.1 The set-up data needed when re-adding an automatically updated lists to the FrameMaker book.

Type of list	**To add**	**Sources**	**Filename Suffix**	**Starting Page Side**	**Main heading tag**
List of Figures	Book File / Add File... / Generated List: List of Figures	FigCaption FigCaptionApp	LOF	Next Available Side	AutoH1NoNum
List of Tables	Book File / Add File... / Generated List: List of Tables	TblCaption TblCaptionApp	LOT	Next Available Side	AutoH1NoNum
List of Listings	Book File / Add File... / Generated List: List of Paragraphs	CodeCaption CodeCaptionApp	LOL	Next Available Side	AutoH1NoNum
List of Items	Book File / Add File... / Generated List: List of Paragraphs	ItemH1, ItemH1App ItemH2, ItemH2App ItemH3, ItemH3App ItemH4, ItemH4App ItemH5, ItemH5App ItemH6, ItemH6App ItemIdentifier ItemTitle ItemTitleChanged ItemTitleRemoved	LOI	Right	AutoH1Chap or AutoH1App
Index	Book File / Add File... / Generated Index: Standard Index	Markers of type Index	IX	Right	AutoH1NoNum

Chapter 5

Delivering the document

5.1 Routine book procedures

Setting the running headers/footers

The running headers and footers throughout the book components are set according to the value of a number of user definable variables containing document meta data.

How to set these variables and how to propagate them to all the component files in the book is explained in the section “Editing the document meta-data” of Section 2.1, "Cover page".

Updating the FrameMaker book

1. From the **File** menu of the book file select **Generate/Update**.
2. Make sure that the file to be regenerated is under **Generate**.
3. Click **Update**.

Saving as PDF

Saving as PDF is straight forward, except for the setting up of bookmarks. It is recommended that the same hierarchy as the FrameMaker table of contents is used. Thus, the list below **Include Paragraphs:** of the **Acrobat Setup** dialog should contain all heading paragraph tags, using the same hierarchy as in the generated table of contents.

For the list of paragraphs tags that you should use as sources of PDF bookmarks, see Chapter 4, ”Automatically updated lists”.

To guarantee that hyperlinks within the document work properly in PDF, it is better to produce one PDF file for the entire book document, as opposed to single ones for each source file.

5.2 Converting to HTML

Both the book* and single file layout templates are pre-configured for saving as HTML, using the built-in **File > Save As... HTML** module of FrameMaker. In both cases, you can easily customize the configuration by setting the values of a few variables that are collected conveniently on the **WWWVariables** reference page, .

3-step conversion procedure

The document-specific customization required when converting a document to HTML is done in three steps:

1. Customize the conversion configuration, by modifying the values of the WWW variables. This is done only once.
2. Doing **File > Save as... HTML** in FrameMaker.
3. After the HTML files are generated, manually do some modifications on the file system. This is done every time that the HTML files are generated.

Step 1: Customizing the conversion, before saving as HTML...

From the book file window, double-click on the *first* listed file (normally `frontmatter.fm`) to open it†. Go to the **WWWVariables** reference page (**View**:**Reference Pages**) on which you will see the list of variables shown in Table 5.1, as well as their current value and a suggested value. To modify, just double click on any of the variables under the **Current Setting** column.

Table 5.1 Pre-defined user variables allowing easy customization of the document conversion to HTML.

FrameMaker Variable Name	Description
WWWAuthor	Full HTML code, inlcuding links if any, for the page signature name(s).
WWWContextLinks	Full HTML code for any links and / or information that will appear at the beginning of all generated pages.
WWWCopyright	Copyright notice. Appears on bottom right of all generated pages.
WWWH1NewPage	“Y” or “N”. If “Y”, then all headings of level 1 trigger a new web page.
WWWH2NewPage	“Y” or “N”. If “Y”, then all headings of level 2 trigger a new web page.

* For the book, the configuration of the HTML conversion assumes that entire books, as opposed to single individual files, are saved as HTML.

† For the single file template, just directly open the file.

Table 5.1 Pre-defined user variables allowing easy customization of the document conversion to HTML.

FrameMaker Variable Name	Description
WWWNavButtonContents	Full HTML code for a button to a general table of contents page. Appears on the right part of the navigation panel on each HTML page.
WWWNavButtonIndex	Full HTML code for a button to a document index page. Appears on the right part of the navigation panel on each HTML page.
WWWNavButtonsFirst	Full HTML code (using also FrameMaker macro buidling blocks), for the left part of the navigation panel that appears on the first (top) HTML page.
WWWNavButtonsLast	Full HTML code (using also FrameMaker macro buidling blocks), for the left part of the navigation panel that appears on the last created HTML. Has no effect if both **WWWH1NewPage** and **WWWH2NewPage** are set to "N".
WWWNavButtonsSubPages	Full HTML code (using also FrameMaker macro buidling blocks), for the left part of the navigation panel that appears on each intermediate HTML page, i.e. not the first and not the last created pages. Has no effect if both **WWWH1NewPage** and **WWWH2NewPage** are set to "N".
WWWInstitutionName1	The full HTML mark-up for the logo(s) and text for the institution(s) name(s). This is what the paragraph **DocInstName** is converted to (on cover page).
WWWINstitutionName2	The continutiona of previous variable, to overcome the 255 character limit of frame variables.

Step 2: Save as... HTML

WARNING!
Due to limitations (and bugs!) in the **Save as... HTML** feature of FrameMaker 5.5.6, **please** follow **exactly** the following steps to convert your document to HTML.

To save a book to HTML, do:

a. Open the book file.

b. Double click on the **first** file in the book (the one listed first in the book file window) to open it.

c. Save the **first** file in the book (**File > Save**).

d. From the book file window, do **File > Save As...**, choosing HTML as format, and preferably placing generated files in their own subdirectory, e.g. `html`.

e. Go back to the **first** file in the book and **close without saving***, or alternatively

*This is to workaround a FrameMaker 5.5.6 bug: if you ever forget this step, the variables on the reference pages will not work anymore. To recover you will simply need to re-import references pages from a fresh copy of the template.

just do **File > Revert to Saved**.

Step 3: File system modifications, after saving as HTML...

After saving as HTML, you need to make the following manual adjustments:

a. Go to the directory containing the generated HTML files, e.g. **`html/`**.

b. Copy the file **`sdlt-master.css`**, provided with the layout template, over the **`.css`** file generated by FrameMaker for your HTML document. E.g. (UNIX):

```
cp -p <template>/sdlt-master.css mydoc.css
```

c. View the generated HTML file(s) with a web browser.

d. If you have set-up the WWW conversion variables such that HTML pages include navigation buttons that link to the global table of contents or to the index, then create symbolic links **`contents.html`** and **`index.html`** that respectively point to the HTML files with the global table of contents and index. E.g. (UNIX):

```
ln -s mydoc.4.html contents.html
ln -s mydoc.1e.html index.html
```

Alternatively, just copy the files:

```
cp -p mydoc.4.html contents.html
cp -p mydoc.1e.html index.html
```

where **`mydoc.4.html`** and **`mydoc.1e.html`** are replaced with the actual names of the generated HTML files for your TOC and index.

e. Create a symbolic link **`welcome.html`** that points to the first file of the generated web. E.g. (UNIX):

```
ln -s mydoc.html welcome.html
```

Or, alternatively, just copy the file in the same way as in previous step.

Chapter 6

Making content templates

Content templates intended to be used in combination with the SDLT must respect a few requirements, specified and explained in this chapter.

6.1 Definition of a content template

A content template is physically composed of up to four FrameMaker files, detailed in Table 6.1. It is recommended that the naming of the files follows the suggested convention in Table 6.1.

Table 6.1 The four file components of a content template for a specific document type, e.g. DOCTYPE.

Recommended file name	Contains
DOCTYPE-frontmatter.fm	Sample material and related filling out instructions for the front matter part. This file may not always be required.
DOCTYPE-chapter.fm	Sample material and related filling out instructions for the (numbered) chapter part.
DOCTYPE-appendix.fm	Sample material and related filling out instructions for the (alphabetically numbered) appendix part. This file may not always be required.
DOCTYPE-guide.fm	A very brief (couple of pages) guide to: a) inform on any special additional steps for combining with the layout template, and b) additional information about how to apply predefined formats in the layout templates to this type of content.

The generic way to combine a content template with the layout templates is explained in Section 1.5, "Combining with the content template". Any additional steps will be documented in the guide for the specific content template.

6.2 The sample material files

The sample material files must respect the following conditions:

a. For each one, the corresponding component layout template is used.

   Thus, a blank copy of the front matter layout component template should be used to *house* the sample front matter. Similarly for chapter and appendix sample material.

b. Any new formats must be amongst the ones specified in Appendix A.2, "Acceptable additions of format tags", and must be documented (Section 6.3).

   Ideally, there should be no new formats or format overrides with respect to the formats defined in the layout templates (see Section 1.7, "No template customizations!"). However, certain content material may require particular layout patterns for which a template extension will be very helpful, e.g. a special kind of table. Fortunately there are several types of extensions that, although they technically constitute a template customization, they are not problematic.

6.3 The guide document for a content template

The documentation file specific to the content template should be small. It should include only that information that is not in this manual but is useful to a user of this particular content template. It must have the following sections:

a. **The files for this content template**

   This section will list all the files (up to four) that make up this content template, with a brief description for each, as indicated in Table 6.1.

b. **Special new format tags for this content template**

   The new format definitions should only be amongst those that specified as acceptable in Appendix A.2, "Acceptable additions of format tags". This section will consist of a table that lists:

   a. The name and type of the new format definition.

   b. Any required additional steps to properly incorporate this format into the combined template. (The generic procedure to combine is provided in Section 1.5, "Combining with the content template".)

c. **Special information blocks**

   This section will consist of a table that lists:

   a. Any information blocks, with a clear shape or pattern, that may typically be present in this kind of content material.

   b. How to format each identified information block using the predefined formats in the combined template.

When writing a guide for a content template, start from an existing one (for example the one for the URD), using it as a template.

Appendix A

Miscellanea

A.1 The distribution file

The distribution file comes in the form of a UNIX tar file, accessible from the following World-Wide Web URL:

`http://framemaker.cern.ch/sdlt/`

Once you download, you can unpack this archive file with **WinZip** (Windows), **Expander** (Macintosh) or, for UNIX systems, with the command:

`tar xvf SSDLT-V2.0.tar`

Listing A.1 The contents of the distribution file for the SDLT .

```
SDLT-V2.0/                          Root directory

    README
    COPYRIGHT
    HISTORY

    doc/
        ugSDLT.pdf                  User's Guide

    sdltbook.tar                    Archive file of sdltbook/ dir,
                                    as starting point for a new
                                    document

    sdltbook/                       Book document subdirectory
        README
        sdlt.bk                     FrameMaker book file
        frontmatter.fm              FrameMaker files...
        sdltTOC.fm
        sdltLOF.fm
        sdltLOT.fm
        sdltLOL.fm
        chapter.fm
        sdltLOI.fm
        appendix.fm
        sdltIX.fm
        sdlt-master.css             Cascading Style Sheet file
        imported/                   To caontain any imported files
        html/                       To contain generated HTML
```

Listing A.1 The contents of the distribution file for the SDLT .

```
    sdltsinglefile.tar              Archive file of sdltsinglefile/
                                    dir, as starting point for a
                                    new document

    sdltsinglefile/                 Single file doc subdirectory

        sdltsinglefile.fm           FrameMaker file
        sdlt-master.css             Cascading Style Sheet file
        html/                       To contain generated HTML

    lib/

        headings-big-appendix.fm    Format files for large style
        headings-big-autofiles.fm   headings
        headings-big-chapter.fm

        headings-small-appendix.fm  Format files for small style
        headings-small-autofiles.fm headings
        headings-small-chapter.fm

        logos.fm                    Reference file, contains logos
```

A.2 Acceptable additions of format tags

Section 1.7, "No template customizations!" provides a list of possible problems with private template customizations. However, not all customizations are dangerous – in fact, a few are more useful than they are problematic. All local format overrides are to be avoided at all costs, but, for **some** format types (see Table A.1) it is OK to add new definitions, i.e. new tags.

Whenever a format is added, the chosen tagname is very important! It is a good idea to prefix all tagnames with the name or identifier of the content-only template for which it added, e.g. URD-tagname. This will keep these formats together nicely in their respective catalog, as well as indicate their use to others.

Table A.1 lists the different FrameMaker formats, organized in four groups, and indicates why adding new tags of these is acceptable or not. It is OK to add format tags only for groups A and B. In all cases, the following must always be well understood and respected:

- Only new tag names can be used, i.e. no changing (overriding) of standard format definitions! A clear naming scheme, as suggested above, is highly recommended.
- All custom added format tags must be maintained separately.

Table A.1 FrameMaker formats may be grouped into the following groups, depending on how they affect management of documents and templates. It is OK to add format tags for the format types in groups A and B.

Format group	Format type	Comments
A (✔)	Table Variable definition Conditional text setting Cross-reference Colour definition	• HTML generation **not affected**. • Updating formats from standard template is not affected. • Instance **can** be copy and pasted into a standard document[a].
B (✔)	Master page Reference page	• HTML generation **not affected**. • Updating formats from standard template is not affected. • To pass to another document, must be explicitly imported as a format, i.e. no copy and pasting of instances.
C (✘)	Math definition	• HTML generation **not affected**. • If standard template contains any math definitions, then updating formats from standard will **destroy** any locally present math definitions.
D (✘)	Paragraph format Character format	• HTML generation **is affected**, and standard HTML conversion rules must be changed. • May necessitate changes in other places, such the automatically updated lists, running headers, etc.

a. Copying and pasting a custom table instance into a standard document does **not** automatically add the table tag to the catalog. However, tags of the four other format types in group A are automatically added to their respective catalog.

A.3 Updating formats from new template

Format definitions of book document files can be updated to newer definitions from a newer release of the templates. There are 2 things to remember:

a. Keep a copy of the user variable definitions in your document, so that you can reset them as they were after you import formats from new template.

b. Update each **type** of document file from the corresponding component template file.

Updating formats in all files in a book document

1. Make a copy of the frontmatter file of your document, not to loese the user variable settings of your document after you import the new formats.
2. Open the book files of both the new template and your actual document.
3. Shift+click the template book **File** menu, and open all files in book (**`Esc fO`**).
4. Repeat previous step to open all files in your book document.
5. For each component template* in the book template (i.e. for each file type) update the document component files of the same type, doing:
   a. From the **File** menu of your document book window, select **Import > Formats...**
   b. Under **Import from Document:** select a component file from the reference template, e.g. **`chapter.fm`**.
   c. Make sure that all formats under **Import and Update:** are selected.
   d. Remove overrides by selecting both items under **While Updating, Remove:**
   e. Make sure that under **Update:** there are only the component files in your document of the same file type as that selected under **Import from Document:**, e.g. all the chapter files of your document.
   f. Click on **Import**.
6. Reset the user variables definitions in your document to what they were prior to importing the new template formats. This you do by:
   a. Open both the copy of, and the frontmatter file of your document.
   b. Go the user variables reference page in both files and reset the values in the newer file to what they had been in the old file.
   c. Close the old file.
7. Import the newly reset user variables definitions to all files in the book. This is done by repeating step 5 above, with the differences:
   a. Under **Import from Document:** select the frontmatter file of your document (make sure no other frontmatter file is open by mistake!).

---

* Due to a FrameMaker bug, each automatically updated file must be open during updating. Thus, before updating the outline, the toc and the index, make sure that they are open. Do not close them until the very end.

   b. Under **Import and Update:**, deselect **all** formats except for **Variable Definitions**.

   c. Under **Update:**, include **all** the files in your document.

8. Regenerate and update your book document.
9. Shift+click on the book **File** menu of your document and save all files in the document (**`Esc fS`**).
10. Shift+click on the book **FIle** menu of both the document and the template and close all files in both books (**`Esc fC`**).

A.4 Splitting a chapter into several files

This may be done as necessary. The only two problems are that:

1. When printed, each separate file will be printed starting on a new page.
2. As only the first file of the chapter will contain the actual chapter heading, this heading cannot be automatically included in the running header of subsequent files for the same chapter.

The second of these problems can be resolved by inserting a marker of Type **Header/Footer $1** at the beginning of the text containing exactly the concatenation of the following pieces of text:

1. The chapter number (if any).
2. If there is a chapter number, then exactly 2 spaces. Otherwise none.
3. The exact chapter heading text.

A.5 Some useful FrameMaker tips

Editing text

Special characters

From **Help**, select **Keyboard Shortcuts**, then **Key map**. Then paginate through the various illustrated keyboards until you see the character you are looking for.

- Note that on PCs this key map is not available. Use the one provided by the system (**Programs > Accessories > Character Map**) instead.
- Full documentation of character sets in FrameMaker is available from **Help > Online Manuals > FrameMaker Character Sets**.

Hard spaces

Control+Spacebar. Forces multiple spaces in a document with Smart Spaces turned on, as well as prevents adjacent words from being broken onto two lines by FrameMaker.

Forced line break

Shift+Return.

Tables

Selecting an entire table column / entire table

Control-double-click in any cell to select the entire column.
Control-triple-click in any cell to select the entire table.

Inserting a TAB character in a table paragraph

Esc Tab.

Working with books

Always access component files through the book file

As a rule of thumb, to open a component file do not use **File Open...** but just double click on the filename in the book window. This way you will always have a consistent global view to the entire document, with quick access to all component files. Also, file settings such as page and paragraph numbering are set at the book level and can thus only be accessed from the book **File** menu.

Opening / Saving / Closing All Files in the Book...

Shift-click on the book **File** menu, and select one of:

- Open All Files in Book
- Save All Files in Book
- Close All Files in Book

Making hypertext links active, or locking / unlocking files

FrameMaker automatically adds hypertext markers to the automatically updated lists such as TOC and the index. These files can be **locked**, making the hypertext links they contain active. This way they can be used as a hypertext navigation tool for the book document. To lock or unlock a file:

1. Make the file window the active window.
2. Type the following 4 keys in succession: `Esc F l k`

Equations

Typing equations à la LaTeX

In a FrameMaker equation, place the cursor where you want to start typing and type "\" followed by the name of the element (as in Latex), then hit `Return`.

Graphics

Turning display of graphics on and off.

If your document becomes slow to edit because of the graphics, you can turn viewing them off completely by selecting **View:Options...**.

Index

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W