NICE Systems

Writer's Style Guide

June 2014

Document Revision Date: June 16, 2014

Contents

1 Working in FrameMaker

- "Creating a shortcut to the NICE templates" on page 7
- "Installing fonts" on page 8

- Directory structure for a guide

- "Setting up a book" on page 8
- "Version numbering" on page 9
- "Document numbering" on page 9
- "Page, chapter, and appendix numbering" on page 9
- "Naming conventions" on page 9
- "Copyright text inset" on page 10

- Conditional text (best practices on using cond text) Jackie
- Color definitions

- "Saving files automatically" on page 11

- Sharing chapters
- Writing for accessibility (point to help chapter. Avoid using side heads in tables. Also BPs on cond text.

Creating a shortcut to the NICE templates

To save time, you can create a shortcut to the folder containing the latest templates so that they're available in FrameMaker when you create new documents.

➢ To create a shortcut to the templates:

1. Open Windows Explorer and navigate to the `TECHPUBS\resources` folder.
2. Right-click the `book_templates` folder and choose **Create shortcut.**
3. Create the shortcut on your desktop and copy it.
4. Navigate to your FrameMaker templates folder. For example:
   `C:\Program Files\Adobe\AdobeFrameMaker11\Templates`
5. Paste the shortcut.

When updates to the templates are available, sync to the latest files in Perforce.

Installing fonts

The FrameMaker templates use two additional font families not included in Windows or Adobe products:

- ITC Giovanni (used for body text)
- MyriadPro (used for headings)

After installing FrameMaker, you must manually install additional fonts used in the templates. If you do not install the fonts, you will see an error message indicating a font is unavailable when opening a FrameMaker file, and PDFs that you generate from the file will not display the correct fonts.

➤ To install the required fonts in Windows 7:

1. In Perforce, sync to the `Fonts` folder in `TECHPUBS/resources`.
2. In Windows Explorer, open the `Fonts/ITC Giovanni` folder.
3. Select the files with `.IMF` and `.PFM` extensions, right-click, and then click **Install**.
4. Open the `Fonts/MyriadPro` folder and repeat steps 2 and 3.
5. Make sure the fonts are installed by choosing **Control Panel > Appearance and Personalization > Fonts**. Scroll through the list of installed fonts and make sure you see entries for Giovanni Black, Giovanni Book, and Myriad Pro.

Setting up a book

If you are making changes to a book for a new release, branch the book's files to the new release directory and then make changes in that new directory. To set up a new book from scratch, copy the book and chapter files from the `TECHPUBS\resources\book_templates\` directory in Perforce to the location where you want the book to reside, rename the files accordingly, and add them to the book file.

After you rename the book, add the table of contents and index using the standard Framemaker procedures, and then generate the files. The easiest way to apply the correct formatting to these files is to now delete them, overwrite them with the TOC and index files from the book template directory (be sure to save them using the filenames in the book file), and then generate them again. This time, the formatting will be correct when the files are generated.

After you have added your files, check the numbering (see "Page, chapter, and appendix numbering" on page 9). Finally, import the variables from `TECHPUBS\resources\book_templates\Variables.fm` into the cover file of the book, make modifications to the variables specifically for that book (such as the BookTitle variable), and then import the variables from the cover file into the rest of the files in the book.

Version numbering

The following terminology describes our software releases:

- Major release—3.0, 4.0, 5.0
- Minor release—3.2, 3.3, 3.4
- Maintenance release—3.2.3, 3.2.4, 3.2.5
- Hot fix release—300100

When typing the version number for a major release, use only one decimal point (for example, 3.0) even though the doc might later be updated for a maintenance release (for example, 3.0.14). In other words, use *n.n* in a major or minor release rather than *n.n*.0.

Document numbering

To track revisions, use the DocumentRevisionDate variable below the copyright section in the cover page. The DocumentRevisionDate variable is a user variable, as opposed to an automatically updated system variable, defined as the date of the release in which this guide was last revised.

NOTE: If you update a guide between releases, change the variable definition to the date you updated the guide.

Page, chapter, and appendix numbering

Page numbering uses Roman numerals continuously through the cover page and TOC, and continues with Arabic numerals on the first page of the first chapter. The first page of the first chapter continues numbering from the previous file (the TOC). For example, if the last page of the TOC is vi, the first page of the first chapter is 7 and continues throughout the remainder of the document. Make sure the pagination in the document footers matches the pagination in the PDF files.

Use Arabic numerals for chapters and uppercase letters for appendixes.

Naming conventions

When naming FrameMaker files, book files, and folders in Perforce, use all lowercase letters and the underscore character instead of spaces. For example, instead of `StyleGuide/OnlineHelp.fm`, use `style_guide/online_help.fm`.

Copyright text inset

For consistency across documents, the cover template uses a text inset located on the master page for the copyright information. In Perforce, the source text is in the `copyright.fm` file in the `resources/book_templates` folder. If a writer updates the inset in Perforce, the copyright text automatically updates when you open the cover page.

The inset import properties are:

- Flow to import: Body Page Flow
- Formatting of imported flow: Retain source's formatting
- Updating of imported flow: Automatic

Working with conditional text

We use conditional text to single-source our text between the PDFs and online help as well as between the different products. Conditional text is powerful but must be applied precisely to avoid formatting problems.

We have the following predefined conditional text expressions:

- Sales PM Guide: use for generating PDFs of ICM guides.
- Service PM Guide: use for generating PDFs of NICE PM guides.
- Sales PM Help/Service PM Help: use for testing the content as it will appear in the ICM and NICE PM help, respectively. RoboHelp uses this same expression for filtering the content as it pulls it from Framemaker.

Conditional text for use with the online help

There have been problems with conditionalized text from Framemaker causing bad formatting when the help is generated. Therefore, before embarking on a major effort to tag your content with the correct tags, do some test compilations of the help in RoboHelp to see how it behaves when you tag the paragraph returns, especially in numbered steps and bullets, to see whether a blank line is added. Also, because of a bug in RoboHelp, it's best not to apply two types of conditional text tags in the same paragraph. For example, if a menu command is different in ICM, instead of just typing them next to each other and conditionalizing them in the same sentence, create a copy of the paragraph and tag each entire paragraph as SalesPM or ServicePM.

For information on applying conditional text, see the Framemaker documentation. For more information on integrating Framemaker files with RoboHelp, see Chapter 8, "Creating Online Help."

Saving files automatically

FrameMaker gives you several preferences for saving and backing up files automatically. Use this section to set your preferences, which stay in effect until you change them

Typically you want to turn on autosave so that if you forget to save files periodically, the autosave file will likely be the latest saved file in case FrameMaker crashes and the recover file is unusable or unavailable.

➢ To set file preferences:

1. Choose **Edit > Preferences**.
2. Uncheck **Automatic Backup on Save**. This prevents backup files from inadvertently getting checked into Perforce and cluttering your folders.
3. Check **Auto Save** and enter `15` in the text box. You can enter another value, but frequent autosaves can interrupt your work.
4. Click **OK**.

NOTE: If FrameMaker does crash, check the timestamps to see which file is newer, the autosave or the recover file (if available). You can then open the latest one and inspect it before overwriting the `.fm` file that crashed.

2 Writing NICE Documentation

This chapter contains the following sections:

NOTE: For information on creating user stories and tasks in Rally, see Chapter 10, "Working with Support Tools."

General writing guidelines

The following general conventions are intended to ensure accuracy and consistency in our publications:

- State the document's purpose in the first chapter.
- At the beginning of each chapter, provide text that introduces the chapter followed by a chapter table of contents. A chapter table of contents is a bulleted list of the Heading1 headings in that chapter. Introduce the bulleted list with the following sentence:

  This chapter includes the following sections:

  Make each bullet a cross-reference to a Heading1 header and apply the Heading & Page cross-reference style.

Question/Issue: Don't have TopOfPage tags anymore. Do we always want the first H1 to appear at the top of the second page? —

- ~~Apply the H1_Heading1_TopOfPage paragraph tag to the first H1 heading in a chapter. This tag forces the first H1 heading in a chapter to appear on the chapter's second page.~~

  NOTE: ~~In rare case where the use of the H1_Heading1_TopOfPage tag causes pagination problems, use the H1_Heading1 tag instead.~~

Question/Issue: What do we mean by body text? What about step intro ¶s?—

- Make sure every heading paragraph is followed by body text. Do not stack headings.
- Write short, simple, and easy to understand sentences.
- Use the active voice. Avoid the passive voice.
- Don't modify the formats in the FrameMaker templates, and don't add new formats unless you have discussed it with the group because they might need to be rolled into the templates and distributed as a read-in.
- Avoid single-sentence paragraphs.
- Do not stack Note paragraphs.
- Place adverbs adjacent to the verbs they modify. For example, instead of "that set of data is removed from the report temporarily," write "that set of data is temporarily removed from the report."

Question/Issue: What do we need to say about split infinitives?

NOTE: Some style guidelines are different when writing for the online help rather than guides. For more information about writing for the online help, see Chapter 8, "Creating Online Help."

Anthropomorphism

Anthropomorphism is the attribution of human motivation, characteristics, or behavior to inanimate objects, animals, or natural phenomena. Avoid using anthropomorphic terms, verbs, and analogies in your writing. For example:

Instead of: "When you log in, the application knows who you are."

Write: "When you log in, the application authenticates you."

Instead of: "A logical node can have multiple personalities."

Write: "A logical node can have multiple server roles."

Instead of: "Computers do not like to be stopped in the middle of a task."

Write: "Stopping a computer in the middle of a task can cause problems."

BookTitle document-specific variables

When creating a new document, create a new BookTitle variable and add it to the `chapter.fm` template file and note the change in the template history file. Use the format "BookTitle-*NameOfGuide*" and the definition "<EI_EmphasisItalic>*NameOfGuide*<Default Font>." Do not include NICE in the name of the guide unless it is already part of the name, such as *Introduction to NICE PM*.

Cliches

Do not use cliches in your writing. For example, avoid expressions such as "to make a long story short," "last but not least," and "the pinnacle of success."

Conditional text

The conditional text tags and expressions in the templates distinguish text that is specific to a product, such as Service PM (now NICE PM) or Sales PM (now NICE ICM), or to a delivery format, such as Help Only.

The conditional text settings and their definitions typically change from one release to another. For the latest information, including how to import them into your documents, see the `Expressions_`*version*`.fm` file in the `resources/book_templates` folder in Perforce. For conditional text setting specific to the online help, see Chapter 8, "Creating Online Help."

Cross-references

Always use the Heading & Page reference format for cross-references. Never apply a cross-reference to a term without indicating to the users where they are going and what they will find when they get there.

The standard sentence format we use is: For more information about [x], see [y].

For example: For more information about the proper documentation review process, see "Documentation review process" on page 148.

In some situations, you can use a shorter form of cross-reference when you don't want to disrupt the flow of action by adding a whole new sentence. For example: You can also use the C_Code (or CT_CodeTable) character format, as described in "Character format" on page 16.

Also, include the introductory article when cross-referencing to another document, for example, "For more information on configuring dashboards, see the *Application Configuration Guide*.

Code

This section describes how to format code in Technical Publications guides.

Line wrapping in blocks of code

In general, don't break URLs, directory paths, and filenames by inserting return characters (because inserting returns "breaks" them). If inserting a return would break the example, simply let the line wrap.

Separate long, fully qualified filenames into two parts (directory and file). For example:

Instead of:

The setting is in the `config/variants/Plain/partitions/None/data/PolicyType.csv` file.

Write:

The setting is in the `PolicyType.csv` file in `config/variants/Plain/partitions`.

Character format

Use the C_Code (or CT_CodeTable) character format for the following:

- Commands. For example, "… the `m cubepop` command."
- File names. For example, "… the `AvayaNextPopJob.properties` file."
- Directory names and file paths. For example,"… in `/com/merced/datacentral`," and "… in the `Configuration` directory."

Gender

Make your writing inclusive, not sexist. Ensure that the use of pronouns does not exclude either sex or stereotype male or female roles. If you must use names, use gender-neutral names (such as "Chris").

Avoid the use of the possessive, if possible. For example, avoid “Tell your manager that you want his support.” Instead, write “Tell your manager that you want support.”

Avoid “he or she.” Rewrite these sentences to use a plural instead of singular pronoun. Do not use “their” unless it indeed refers to a plural antecedent.

Where appropriate, recast the subject and verb, or antecedents and pronouns, as plural.

Jargon

Use jargon if the technical word or phrase expresses a precise idea that a common word or phrase cannot express. For example, in “The application persists the object to the database...,” write “saves,” not “persists.” Also, in “Use metadata XML files to describe the shape of your data...,” the words “shape of your data” are vague. Consider the possible difficulties of translating jargon.

Page breaks

You can adjust the pagination settings on a paragraph to create a page break. To set a page break, click in the paragraph you want to move and press ESC J SHIFT+C. This is the keyboard shortcut for setting a paragraph to start at the top of the next column.

Specific words and terms

This section describes specific words and terms.

a vs an

When using the acronym NPM, precede it with the indefinite article “an” rather than “a.” For example, “An NPM installation requires…”.

above and below

Do not write “above” or “below” when referring to information that precedes or follows the current sentence. For example, do not write “The example above shows...” or “The example below shows...” Instead, write “previous” or “following.” For example, write “The previous example shows...” or “The following example shows...”

accessible

Question/Issue: New entry. Discuss.

Do not use instead of “simple” or “available.” Reserve this term and “accessibility” to refer to things that all people, including those with disabilities, can easily use.

after vs. once

Don't use "once" when you mean "after." For example, instead of, "Once you get your certificate, do X," write, "After you get your certificate, do X."

appears

Use instead of "is displayed." For example, instead of, "The window is displayed," write, "The window appears." See also **displays**.

application

Use instead of "system" when specifically referring to the product. For example, "This workspace appears on its own tab in the NPM application for users with appropriate privileges."

Use "system" only when writing generic text such as, "a system-generated reference."

as vs. because

Do not use "as" when you mean "because." For example, instead of, "Status metrics are useful as they allow at-a-glance feedback," write, "Status metrics are useful because they allow at-a-glance feedback."

at vs. in

Question/Issue: Need to discuss US convention of "in release x" and "with release x" vs UK convention of "at release x."

auto-complete selector

The official name of a field that suggests potential results when a users begins entering text.

axes

Plural for *axis*. The possessive of *axes* is *axes'*. The axes are known as rows, columns and corners. Previously they were x (columns), y (rows), and z (corner).

axis

Singular. The possessive of *axis* is *axis's*.

back up

Verb.

backup

Noun, adjective.

blast and reload

Use “delete and reload” instead.

See “delete and reload” on page 22.

Bookmark button

The name of the icon in NICE PM that creates bookmarkable links. 

Boolean

In running text, use a capital B. The word *Boolean* derives from the name of George Boolean, the 19th century mathematician. Do not write *boolean* unless the word is capitalized that way in code, for example: `<type class="boolean">`.

breadcrumb

One word. A series of links at the top of a web page that shows the path from the home page of the site to the current page.

buttons

See “UI controls” on page 42.

calendar control

Official name for the calendar icon. Not “calendar chooser” or “calendar pop-up.”

can vs. may

Avoid overusing the phrase “you can” when it is not warranted.

Use *can* instead of *may*, but avoid excessive use. For example, instead of writing “You can use the tree structure to...,” in conceptual paragraphs, write “You use the tree structure to...” In procedures, write “Use the tree structure to....”

Don’t use *may* when you mean *might*. For example, if you write “Port 25 may have conflicts...,” it appears that such conflicts are permissible. Instead, use the verb *might* to suggest the potential for conflicts, which is the idea you really intend to express.

cancelling dialog boxes

Assume that users know how to manipulate dialog boxes. Don’t tell users to click **Cancel** unless it’s a necessary step to return to where they were.

cbean

Correct spelling, capitalized (at the beginning of a sentence only) as *Cbean*. Do not use *CBean*.

Cbuild

Correct spelling and capitalization.

check

Do not use when writing about using check boxes. Instead, use *select* and *deselect*.

check box

Two words. Not *checkbox* or *check-off box*.

checklist

One word. Not *check-list*.

choose

Referring to menu (or submenu) actions, use the verb *choose* instead of *select*. For example, "Choose **Edit** > **Undo**."

When referring to other actions, use the verb *select*, for example, "Click the radio button to select the category that contains the report you want to run."

click on

Do not use. Write *click*. For example, instead of "Click on **Save**," write "Click **Save**."

command

Use *command* instead of *tool*, *command tool*, *script*, or *program*. Also note that we tell users to *enter* commands, not to *run* or *execute* them.

company names in examples

If possible, do not include the names of other companies in examples.

As needed, use company names to identify other companies' products, such as Microsoft Excel.

computer

Not *machine* (except when describing the device used for ETL).

computer names in examples

If you need to use the name of a computer in an example, use a generic name (myserver.mycompany.com) and not the name of your own computer or a release computer. For example:

```
https://myserver.mycompany.com:81/ns/
```

contractions

The following contractions are acceptable:

- aren't
- can't
- didn't
- doesn't
- don't
- it's
- there's
- won't
- you're
- you've

Avoid *you'll* and *you'd*. For example, write "To find a product…" instead of "If you'd like to find a product…."

CSG

The Client Services Group at NICE. Spell out on first reference; use the abbreviation on subsequent references.

CubePop

The process that populates the OLAP data mart. Write *CubePop* when referring to a particular process or job. However, write "`cubepop` command" when referring to the command.

data

Data is always singular: Data *is*, not *are*.

database

One word, not *data base* or *DB*.

data mart

Two words. Not *datamart*.

default configuration

In references to an *out-of-the-box* configuration, write *default configuration*. Write *default configuration* rather than the following:

- The standard configuration
- The default NICE configuration
- The initial configuration
- An initial configuration

For example, "In the default configuration, the filtering is done by the client."

definitions

On first use, provide definitions of unfamiliar terms. Don't define terms for third party applications. For example, don't define application server, web server, or database terms.

Use the *EI_EmphasisItalic* character tag on the first use of the defined term.

delete and reload

Use to describe one of the changed data capture (CDC) techniques used in NICE PM. Do not use *blast and reload*.

desire and desirable

Use *desired* only in procedures, such as, "Choose the desired option." Do not use *desire* in place of *want*, such as, "If you want to view all the options available for a feature at a higher level, click the associated tab."

dialog box controls

Use the term *control* when a dialog box is a single-purpose item containing only one setting. For example, "The axis disappears from the report and is replaced by the Add axis control."

dialog box options

Use the term *options* when a dialog box contains many settings.

dimmed

Use when referring to interface items that are unavailable, as recommended by the *Macintosh Human Interface Guidelines*. Do not use *grayed out*.

displays

Generally, write “The window appears” or “The window is displayed.” Use *appears* instead of *displays* because *appears* does not require an object. However, use *displays* when referring to the application displaying a window, such as “NICE ICM displays the window.”

do

Avoid using *do* as a verb. For example, instead of “to do an upgrade,” write “to upgrade” or to “perform an upgrade.” Similarly, instead of “to do configuration,” write “to configure” or “to create configurations.”

document name

When referring to the document, write “this document” rather than “this manual,” “this guide,” “this book,” or some other term. For example, “This document provides an overview of the application’s architecture.” When referring to another document, use the document variable. Don’t type the document name, because the names often change.

double-click

Not *double click*.

download to

Not *into* or *onto*.

drill down

To go to the next lowest level of a hierarchy. Two words. Avoid the noun form *drilldown*.

drill path

Two words. Not *drillpath*.

drop-down list

Noun. Use when referring to a drop-down controller on a field that displays a list of items the user can choose from to populate the field. Not *pop-up list* or *pull-down list*.

drop-down menu

Noun. Use when referring to a drop-down controller on a menu field that displays a list of menu item choices, for example “choose **Save** from the **File** drop-down menu.” Not *pop-up menu* or *pull-down menu*.

e.g. and i.e.

Avoid these abbreviations. Instead of *e.g.*, use *such as* or *for example*. Instead of *i.e.*, use *that is*.

See also "etc." on page 24.

ellipsis

An ellipsis (three periods) indicates an omission within a quoted passage or the name of a menu command that opens a dialog box. When referring to a menu command that contains an ellipsis, do not include the ellipsis. For example, instead of, "Choose **Find...** from the menu," write, "Choose **Find** from the menu."

em dash

Use an em dash (—) to set off parenthetical material. Do not use two hyphens. Do not enter a space before or after the em dash. To create an em dash in FrameMaker, press CTRL+Q, SHIFT+Q. An em dash sometimes substitutes for a comma, a colon, or parentheses, for example, "This is your mandate—sell the product!"

Question/Issue: Think we need to revisit using an em dash instead of a comma, colon, or paren.

email

Lowercase, no hyphen. Use *Email* in a heading or title.

enable vs. let

Don't use *enables you* when you mean *lets you*. Enable means to turn something on or off.

end user (n.) and end-user (adv.)

In end-user documentation, when writing about the people who use our products or services, write *user* or *users*. Don't write *end-user*, *end-users*, or *employees*.

Question/Issue: ICM docs can't follow this rule because users has a narrower meaning than the people who use our products.

etc.

Avoid this abbreviation. *Etc.* stands for et cetera, meaning *and others*. Use *and so on* instead.

See also "e.g. and i.e." on page 24.

file extensions

For sentences that refer to file extensions, include all the periods, even the period at the end of the sentence, for example, "JSP and HTML files must have the extensions `.jsp` and `.htm`."

When referring to a file type generically, use the file extension in all caps and without the preceding period. For example, when discussing text files, use *TXT files*, not *.txt files*. Similarly, for zipped files, use *ZIP files*, not *.zip files*.

file type

When referring to a file type generically, use the file extension in all caps and without the preceding period. For example, when discussing text files, use *TXT files*, not *.txt files*. Similarly, for zipped files, use *ZIP files*, not *.zip files*.

filename

Noun or adjective. Correct spelling.

The first time you discuss a file, use the whole path, and on subsequent uses you can simply refer to the filename. Additionally, if you are talking about several files in the same location, you can usually refer to the whole path only once. However, note that paths cannot hyphenate, so use your best judgement to avoid large gaps of white space at the end of the line. For example, you might work around a gap by saying "`file.htm` in the `aaa\bbb\ccc` directory" instead of "`aaa\bbb\ccc\file.htm`".

filters

In NICE PM, filters are *applied* and *removed* from reports, not *added* and *deleted* or *disabled*.

first person

The first person singular is the pronoun *I*. The first person plural is the pronoun *we*. Avoid the first person singular. Try to avoid the first person plural. Do not write *we* in the preface to a document.

following vs. the following

When introducing a list or example, use either "Following..." or "The following..." For example, "Following is an example..." or "The following is an example..." Whichever style you choose, use it consistently throughout the document.

free-form

Always hyphenate when used as an adjective. For example, *free-form text*.

future tense

Avoid if possible.

grain vs. level

In NICE PM, dimensions consist of *grains* and hierarchies consist of *levels*.

grayed out

Do not use when referring to interface items that are unavailable. Use *dimmed*, instead, as recommended by the *Macintosh Human Interface Guidelines*.

home page

Correct capitalization when referring to the home page.

i-button

The official name of the information icon on report screens.

icons

See “UI controls” on page 42.

in order to

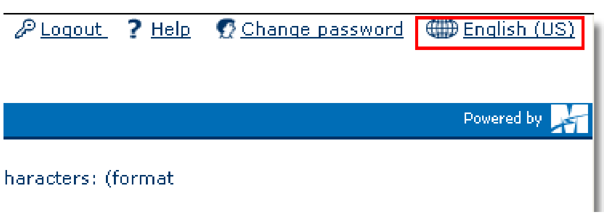
Avoid using this phrase; use *To* instead. For example, instead of “In order to publish the configuration, write “To publish the configuration.”

landing page

When describing administration and configuration of the application, use *landing page* to describe the location of content items on the main screen of a tab in the content tree. For example, “To place a free-form text item on the main Actions screen, place it on the Actions landing page in the content tree.”

language selector

The name of the link on the welcome bar in all screens when the Global Edition is installed.



level vs. grain

See “grain vs. level” on page 26.

if…, then…

Where possible, delete *then* from the concluding clause, for example, "If you are an experienced user, try the Organizer," instead of "If you are an experienced user, then you might want to try the Organizer."

If you want to…

Avoid. Write "To [perform a task]...," for example, "To save the edits, click **Save**."

is displayed

Avoid the passive voice. Write *displays* (with a direct object) or *appears* (does not require a direct object).

left navigation pane

The name of the table of contents-like pane on some screens.

like

Avoid. Use *such as* instead. For example, instead of "... any restrictions put on items like reports," write "... any restrictions put on items, such as reports."

lists

Following are guidelines for lists:

- Introduction—Always introduce a list with a sentence, if possible.
- Parallelism—Express list items in a parallel manner (all fragments or all sentences and same grammatical structure, for example, all gerunds, or all nouns).
- Colon—Use a colon to introduce a list, a series, and indented information. Also, *the following* and *as follows* require colons if a list directly follows one of these expressions. If the introductory statement containing *the following* or *as follows* is a complete sentence (with a period) and is followed by one or more complete sentences prior to the list, use a colon immediately prior to the list.
- Period with fragments—Don't use a period after single word items or fragments in a list.

log in/login

Use *log in* as a verb. Use *login* as a noun or adjective. For example, "Log in to the application" and "Type your login name."

In general, use *log in* and *login* for applications. Use *log on* and *logon* for networks and servers. However, if the user interface does not use this standard, use the same terminology as the product.

log off (v.)

Verb. Not *logoff*.

log on vs. logon

Use *log on* as a verb. Use *logon* as a noun or adjective. For example, "Log on to the server" and "Type your logon name."

log out vs. logout

Use *log out* as a verb. Use *logout* as a noun or adjective.

machine

Use *computer* instead. (The exception relates to the device used for ETL.)

make sure vs. ensure

Use *make sure.*

Manage Content Tree icons

The names of the icons on the Manage Content Tree screen are:

- Create tool 
- Add tool 
- Move tool 
- Remove tool 

may

Avoid. Often confused with *might*. To indicate permission, write "You can..." or "You might..." rather than "You may...."

menus

Use the verb *choose* instead of *select* when referring to menu (or submenu) actions. For example: "Choose **Edit** > **Undo**."

When you're referring to an individual menu item, it's a command, for example, "the **Undo** command on the **Edit** menu."

If a command name or dialog box option ends with a colon or ellipsis, do not include this punctuation.

Use the greater than symbol (>), also called an angle bracket, between the menu and command name, for example, "Choose **Edit** > **Undo**."

NICE

Use when referring to the company informally, such as "NICE recommends that you...."

metabean

Use *metabean* as one word in lower case letters. Do not use *MetaBean* or *metaBean* unless you are copying code or reproducing a command.

metabean type

Use *metabean type* as two words in lowercase letters.

migrate vs. upgrade

Verbs. Use *upgrade* when you write about applications; for example, "You upgrade the application."

Use *migrate* to refer to updating the application's data and configuration information.

name/value pair

Not *name-value pair*.

NICE Incentive Compensation Management

Use the full term when referring to the product for the first time. Use *NICE ICM* for subsequent references in the same document.

NICE Performance Management

Use the full term when referring to the product for the first time. Use *NICE PM* for subsequent references in the same document.

NICE Systems

Use when referring to the company.

numerals vs. words

Use numerals for 10 and greater. Spell out numerals from zero through nine if the number does not precede a unit of measure or is not used as input. For round numbers of 1 million or more, use a numeral plus the word, even if the numeral is less than 10.

Spell out numbers that begin a sentence. For example, instead of "100..." write, "One hundred..."

Observation Manager

Not *Observations Manager.*

OLAP and OLTP

Abbreviations for Online Analytical Processing (OLAP) and Online Transactional Processing (OLTP). Assume your audience knows these terms, so use the abbreviations for first and all subsequent references.

onscreen and online

Don't hyphenate these terms.

online help, online Help, help, or Help?

**Question/Issue: Which do we want to use as the standard? MS says: In general, avoid *online Help*; just use *Help*. However, *online Help* and *online Help files* are acceptable when necessary to describe the Help system.
(I think we want to capitalize it as *Help*, no?)**

operations centers

When referring generically to the offices that oversee call centers, use the term *operations centers*, not *operation center*, *operation centers* or *ops centers*.

Question/Issue: That's odd—we seem to have moved away from this term and use *call centers* now exclusively.

page

Avoid. Use *screen* to describe the user interface, except when referring to the home page.

parentheses

Do not use parentheses to create an optional plural noun, such as *member(s)*. Instead, write out "one or more members."

pathname

Not *path name*. Also *path* as in "Enter the path for your home directory."

permissions

Use only when referring to specific features that have permissions set through a permissions user interface, such as dashboards.

When referring to the concept of permissions in a general, conceptual way, use the term *role-based access*. For example, "When you log in to NICE PM, the application authenticates you and displays content according to your role-based access."

please

Avoid using *please* in the documentation and the user interface, including messages.

plugin

Noun. No hyphen. Not *plug-in*.

pop-up menu

Use *drop-down menu* or *drop-down list* instead.

re-

Hyphenate *re* as necessary to avoid confusion with another word. For example, it is necessary to hyphenate *re-cover* (to cover again), as opposed to *recover* (to regain).

read-only

Correct spelling.

read/write

Noun. No hyphen. Not *read-write*.

Reset button

The official name of the icon that resets reports to their original state. 

right-click

Correct spelling, for example, "Right-click the mouse and choose **Copy** from the shortcut menu."

say

Do not use; too informal. Use *specify* or *indicate*.

screen

Not *page*. The applications display screens, not pages.

serial commas

Use a comma before the conjunction. For example, “red, white, and blue” NOT “red, white and blue.”

set up

Verb. For example, “I am going to set up my home computer.” As needed, be more specific, for example, use the terms *load*, *configure*, and so on.

setup

Noun. For example, “What type of setup (meaning, arrangement) should we use when we place the props on the stage?”

Adjective. For example, “What setup characteristics influence the design of the stage?”

Do not use when referring to the application or a customer’s specific instance of the application. For example, instead of “your company’s setup,” write, “your application” or “your configuration.”

should vs. must

Question/Issue: Think we need to address these terms. I avoid *should* because it’s imprecise. Here’s what MS says: Use *should* only to describe a user action that is recommended but optional. Use *must* only to describe a user action that is required. [I think we use *should* to describe something more vague: “This system should have the same topology as your production environment.” This use of *should* can frustrate users who wonder what it means if something else is true.] MS goes on: Avoid using *should* to indicate probability (such as, “You should receive a notification within 24 hours.”) It is all right, and often better, to use alternate ways of specifying recommendations or requirements. For example, use the imperative mood. For optional actions, you can use a phrase such as “we recommend,” but avoid “it is recommended.”

since vs. because

Don’t use *since* when you mean *because*. Use *because* to refer to a reason and *since* to refer to a passage of time. If it is possible to misinterpret the meaning of *since* as referring to a reason, rewrite the sentence.

single sign-on

Spell out the term on all uses. Not *SSO*.

some

Don’t use.

such as

Use instead of *like*.

See “like” on page 27.

swap icons

Formerly called “pivot buttons,” this is the official name of the icons used to switch axes in a report.

system

Use only when writing generic text, such as “a system-generated reference.” When specifically referring to NICE PM or NICE ICM, use “application” or “instance” instead.

tabs

Use *tab* in procedures, such as “On the Admin tab, click **Employee Manager**.” Use *screen* when referring to the content of a tabbed page generally, such as “On the Admin screen, you can access the Employee Manager.”

In general, do not apply the GO_GuiOption tag to the names of tabs. For more information, see “UI labels” on page 44.

that

Avoid unnecessary uses of the word *that*. For example, write “An expense card is a corporate credit card you use to pay for business and travel expenses you incur...” instead of “An expense card is a corporate credit card that you use to pay for business and travel expenses that you incur....”

that vs. which

See “which vs. that” on page 35.

this

Avoid using *this* as a demonstrative pronoun, for example, “The foo feature is enabled in the default configuration. This allows you to...” Instead, use *this* as an demonstrative adjective, such as “This feature allows you to...” or combine the sentences, such as “allowing you to...”.

time stamp

Noun. Not *timestamp*.

time-stamp

Verb.

traffic lights

Not *streetlights* or *stop lights*.

uncheck

Do not use when writing about working with check boxes. Instead, use *deselect* and *select*.

upload to

Not *onto* or *into*.

URL

Written in all caps.

username

One word.

via

Avoid this Latin word. Use *through* or *by way of*.

we

Do not use *we* to mean NICE, such as "we recommend...." Instead, use "it's good practice to," "as a best practice," or state what you want the user to do. In some cases, you might find it appropriate to say *NICE*.

web

Correct spelling and capitalization. However, when referring to the World Wide Web specifically, use *Web*.

web browser

Correct spelling and capitalization.

website

Correct spelling and capitalization.

where vs in which or on which

Use *where* instead of *in which* or *on which* when describing where a program or component resides. For example, instead of "Navigate to the directory in which the installation program resides," write, "Navigate to the directory where the installation program resides."

Or instead of, "Specify the name of the computer on which the HR system is running," write, "Specify the name of the computer where the HR system is running."

which vs. that

Use *that* for clauses that are crucial to the identification of the phrases they modify (restrictive) and are *not* set off from the rest of the sentence by commas. For example, if there are several tabs, and you want to clarify that you're talking about the one the administrator uses, you would say "The tab that the administrator uses is on the right."

Use *which* for clauses that add to the knowledge of the phrases they modify without being crucial to the identification of the phrases (non-restrictive) and *are* set off from the rest of the sentence by commas. In the same example, if there is only one tab, and you simply want to provide more information about who uses it, you would say "The tab, which the administrator uses, is on the right."

workflow

One word.

workforce

One word.

workstation

Noun. One word, no hyphen.

would like to

Avoid. Use *want* instead. Do not use *desire* or *wish*.

X

When referring to the use of the character x in a table or chart, use the term *exes*. For example: "NICE PM provides six icon sets: traffic lights, checks-and-exes, smiley faces…."

Abbreviations and acronyms

Avoid the excessive use of abbreviations and acronyms in running text. If you use them, spell them out on first usage, and then place the abbreviation or acronym in parentheses directly following the first usage, for example, "While viewing a report that measures your average handling time (AHT)..."

Familiar abbreviations can be used if they are explained when they are first used in text, for example, "The central processing unit (CPU) is the brain. The CPU does the work of...."

Add an "s" to an abbreviation or acronym to form the plural form. Do not use an apostrophe or (s).

For abbreviation guidelines specific to the online help, see Chapter 8, "Creating Online Help."

Alphabetization

In indexes, entries are alphabetized word-by-word (meaning spaces are considered in the sort). For example, *data mart* appears before *database*. If we want letter-by-letter alphabetization in the indexes, we can add FrameMaker's various spaces to the IgnoreCharsIX paragraph on the index template's reference page.

Bulleted lists

Use a bulleted list to present an unordered list of items or a set of points.

- Pagination—Manually verify that the pagination of bulleted lists is correct. Specifically, a bulleted list should be on the same page as the paragraph that introduces the list, and one item of a bulleted list should not appear on a page by itself. It's preferable to keep the bulleted list on one page. However, if the bulleted list is long, you can split the bulleted list so that at least two bulleted items appear on each page.
- Periods—Use periods at the end of the of a bulleted item when the bulleted item is a complete sentence, otherwise you can omit periods. However, if at least one item in the bulleted list requires a period, then all items in the bulleted list should have a period, even if the other items are not complete sentences.
- Introductory phrases—If a bulleted list has introductory phrases, it can have any combination of character tags, em dashes or colons, depending on the topic. For example, if the bullet describes a button or field on the user interface, use the GO_GuiOption character tag followed by a colon. If the introduction is a piece of code, use the C-Code character tag followed by an em dash with no spaces around it. If the bulleted item describes a generic

concept or feature, or is a list of cross-references, then the character tag is regular body text (default) with an em dash. To emphasize the introductory term, use EB_EmphasisBold. Following are some examples:

- **Name:** In this field, type the name of the agent being evaluated…
- `m_cubepop`—This is the command for running a cube population…
- File Sniffer—Among the options are to use a file sniffer…
- **Create users**—Each person in the org chart can have a user name…

- Em-dashes—When using em dashes in a bulleted list to separate introductory phrases from their descriptions, there should not be space on either side of the dash. In FrameMaker, the keyboard shortcut for em dashes is CTRL-Q SHIFT-Q.

Callouts

A callout is a written label that identifies a specific part of a graphic in an anchored frame. Use the FIC_FigureCallout character tag for single-line callouts and the FIC_FigureCallout or FIC_FigureCalloutCenter paragraph tag for multi-line callouts. Use an initial capital letter for the first word of the callout. Do not use a period at the end of the callout unless it is a complete sentence. For more information, see "Adding callouts to screenshots" on page 52.

Capitalization

In headings, use initial caps (only capitalize the first word in the heading). The only exception is the chapter title: use title caps (all words capitalized except for prepositions) for chapter titles.

In body text, only capitalize proper nouns (trademarked product names, proper names, place names, and proper time indicators) and the first word in the sentence.

Code samples

This section describes the guidelines to use when documenting code. It contains the following topics:

- "Formatting individual items" on page 37
- "Formatting blocks of code" on page 38
- "Specifying standard directory names" on page 38
- "Formatting commands and filenames" on page 38

Formatting individual items

This section describes how to format individual code items, such as packages, modules, attributes, elements, parameters, filenames.

Packages, modules, cbeans, nodes, and files

A number of elements are represented both as common nouns (for example, property type) and components of the data model (for example, `propertyType`). Note the differences in capitalization, word spacing, and typography. The following guidelines apply to these elements:

- Use the CO_Code character tag for the closed-space, initial-capped form.
- Use the closed-space, initial-capped form when you refer directly to a file name, field name, or data model component. For example:
    - Notice the cbean definition of the decision node.
    - Assume you want to create a decision node that routes the workflow to the `HighAmountAssignment` node.

Question/Issue: TBD

Quotes

When you copy and paste code samples into FrameMaker, single and double quotes are not converted to curly quotes. However, when you type a code sample, the smart quotes feature converts them to curly quotes as you type. You must replace curly quotes with straight quotes in code samples.

➢ To type a straight quote in FrameMaker:

- ESC SHIFT+″ creates a single straight quote (")
- ESC CTRL+′ creates a straight apostrophe (')

NOTE: To search for straight quotes, type `\x22` in the **Find/Change** window. To search for straight apostrophes, type `\x27` instead.

Formatting blocks of code

This section describes how to format blocks of code.

Question/Issue: TBD

Specifying standard directory names

This section describes conventions for specifying directories that are part of NICE PM and NICE ICM.

Question/Issue: TBD

Formatting commands and filenames

Question/Issue: TBD

Headings

Headings should be sentence capitalized, except for chapter titles, which should be title capitalized. Be consistent in wording headings, and strive for economy. Headings should not exceed one line because of template restraints and the odd effect it produces in the table of contents.

In creating headings for documents, use gerunds in sections that discuss a task, for example:

Creating a Test Exchange

For numbered procedures, use the infinitive, for example:

To Create a Test Exchange:

In sections that discuss a feature or component, you can use the form "About broadcast protocols" or try to use a gerund to keep the heading parallel with others in the chapter:

Understanding Broadcast Protocols

Working with Broadcast Protocols

In conceptual sections, generally use the plural of the noun unless the singular is correct. In most cases, use the singular for task headings. For example:

Adding Users

To Add a User:

Understanding Broadcast Protocols

Creating a Test Exchange (you'd only create one)

About the Catalog Hierarchy (there's only one)

Notes and warnings

Following are guidelines on using notes and warnings.

- Use a note to break out related, reinforcing, or other special information.
- Keep your note short and relevant.
- Make sure you write notes consistently so that a reader learns to recognize them.
- Never use a note to cite safety information.
- You must use a warning whenever you describe a situation that has a risk of irreversible destruction to data or software.

Here is an example of a note and its format:

NOTE: Before using a note, try to work the info into the text, unless you really need it to stand out or if it would otherwise result in a single-line paragraph.

Procedure writing

This section presents guidelines for writing procedures.

General guidelines

When writing procedures, follow these general guidelines:

- Before a procedure, pull out information that the user should know before starting the procedure.
- Try to write procedures that can be reused and cross-referenced later. For example, if there is a procedure that takes a few steps and must be repeated several times throughout the document, you could write that procedure separately, then refer to it when writing additional procedures.
- Try to make each step a single sentence. If there is further explanatory text or examples you need to add, add it in a new non-numbered paragraph using one of the Body paragraph tags.
- You do not need to include "clean-up" instructions, such as instructing the user to close or cancel the dialog, unless it's required before going on to the next step or for applying the changes or making them display.
- Don't use the words "First," "Second," and so on, in the text of a step.
- Always document the preferred way of performing an operation. Don't feel you have to document all possible ways of performing the same task.
- When referring to another step, write step #, not Step #, for example, "This action closes the HTTP connection established in step 1. Repeat step 4."
- Only number steps that the user takes. Do not number text describing what happens next. For example, don't number "The page you choose appears."
- If the steps of a procedures become too bulky because of important explanatory information for each step, pull that information out into the paragraph before the start of the procedure.

Optional and repeated steps

If a step is optional, write "(Optional)" at the beginning of the step. For example:

1. (Optional) Enter a comment for the item.

If a user can skip several steps, consider breaking out the steps into a separate procedure. If there are just a few steps the user can skip, use the word “skip.” For example:

1. If you are using SAP or PeopleSoft, skip to step 2. Otherwise, do the following:
    - Run Oracle.
    - And so on.

To instruct the user to repeat a series of steps, consider breaking out the steps into a separate procedure, or tell the user to repeat the steps, for example, “Repeat steps 3 through 6 for each variant.”

Proper names

Capitalize proper names and avoid using *the*. The following words are proper names:

- File Sniffer
- NICE Performance Management (“NICE PM” on subsequent references)

Tables

The following are general guidelines for using tables:

- Keep information in table cells short. If you have more than one paragraph per cell, consider using headings and paragraphs instead.
- Try to keep tables from spanning more than two pages. Tables that span many pages can be difficult to follow.
- Do not use table titles for tables that do not span pages. For tables that do span pages, use the LOT_LinedTitled style and the Table Continuation variable at your discretion.
- Don’t use back-to-back tables without some explanatory information between them.

UI controls

Question/Issue: Here's a table I'd like to add here for quick reference. Good idea?

Element	Tag	Interaction	Comments
Button	GO_GuiOption	Click	
Check box	GO_GuiOption	Click	To describe states, use "checked" and "unchecked."
Close button	GO_GuiOption	Click	
Collapse arrows	Default ¶ Font	Click	
Content item	Default ¶ Font	—	
Dashboard	Default ¶ Font	—	
Dialog box	Default ¶ Font	—	
Drop-down list	Default ¶ Font		
Drop-down menu	Default ¶ Font		
Expand arrows	Default ¶ Font	Click	
Info icon	Default ¶ Font	Click	
Maximize arrows	Default ¶ Font	Click	
Menu	GO_GuiOption	Choose	
Minimize arrows	Default ¶ Font	Click	
OK button	GO_GuiOption	Click	
Page	Default ¶ Font	—	Lowercase
Pane	Default ¶ Font	—	**Question/Issue: Do we need two terms?**
Panel			
Radio buttons	GO_GuiOption	Click	
Scrollbars	Default ¶ Font	Click or drag	
Tables	Default ¶ Font	—	
Tabs	Default ¶ Font	Click or display	Lowercase.
Text box	GO_GuiOption	Enter	Not type

Use the **GO_GuiOptions** character tag when referring to user interface controls that you interact with (buttons, tabs, menus, and menu choices). Do not include colons and ellipses. For example, write "Choose **Edit permissions** to change the report's permissions" instead of "Choose **Edit permissions...** to change the report's permissions." If you do not interact with a control (such as a section or icon), you use the default character format and do not apply **GO_GuiOptions**. Following is more information about specific controls.

- Buttons and icons—If an icon performs any kind of function when clicked, it is called a button. For example, the Reset button is an icon in the header of a report that resets the report to its original state when clicked. Icons that do not perform a function when clicked are called icons. When instructing the user to click a button, simply use "click" with the button name, such as "Click **Reset**" instead of "Click the **Reset** button." The only exception to this rule is if there are two UI elements with the same name, requiring you to differentiate between the New button and the New link, for example. In this case, please encourage Engineering and Product Management to redesign the UI.

- Dialog boxes—Use the Default Font character style, and mirror the spelling in the user interface, for example, "In the New User dialog box, type...."
- Fields—Do not bold the name of a field unless it's something you can actually interact with. For example, don't bold the Foo field but do bold the New button: "Next to the Foo field, click **New**."
- Menus—The correct verb is "choose," such as "Choose **File > New**" or "Right-click the user's name and choose **Properties**."
- Navigation bar—Refer to the navigation bar on the left as the navigation bar, not the navigation pane.
- Operations—Describe operations as follows, "Click **Save**," not "Click the **Save** button" or "Click the **Save** link." Do not write "Select **Save**."
- Tabs—When a window or dialog box has tabs, write "Click the User tab." Do not write "select," "choose," or "highlight" when discussing tabs.
- Text—To instruct the user to enter text, write "Type a description up to 130 characters in the Description text box." For a combo box, where you can enter the text or select from a drop-down list in the same control, use the following convention: "Select the value from the Max % drop-down list or type the value into the field." If there is an option to type a value or click a browse button, you do not need to spell out the usage of the browse button unless it uses an unintuitive icon. For example: "In the Data Port Group field, type a unique name (up to eight characters, no spaces) to specify a new group, or browse to an existing group." To describe highlighting a region of text, use "highlight," not "select."
- Window—The application window is typically broken down into "tabs," which contain a "navigation bar" on the left and a" content area" on the right by default. The navigation bar and content area may be further divided into "sections." You do not use the **GO_GuiOptions** style to refer to these elements unless you're referring to clicking a tab. For example, "On the Admin tab, click **Save**." Also, use these terms sparingly and only as needed. For example, write "Click **Properties** in the navigation bar" or, if Properties is somewhat hidden, write "In the Options section of the navigation bar, click **Properties**." You should rarely if ever refer to the window (unless you're describing customization of the application window, for example) or content area. For example, instead of writing "Click **Details**. The Foo tab changes to show the Details view" you should write "Click **Details** to display the contact details" or "Click **Details**. The contact details appear." Do not use "screen" or "page."

UI labels

In general, when referring to UI labels, you don't bold the names of any UI elements unless you're instructing them to click or choose them. The rule of thumb is that you either bold an element's name OR you leave it in plain text and follow it with its noun (field, link, section, etc.). Typically, you bold the names of buttons, check boxes, drop-down lists, and tabs when you're clicking them. For example:

Next to the Foo field, click **New**. (Do not write "click the **New** button", as the bold text makes the noun unnecessary).

In the File menu, choose **Options > Properties**.

Choose **File > New**.

On the Admin tab, click **New**.

Click **Admin**, and then click **New**.

The exception to this rule is when there are multiple UI elements with the same name or it would be hard to find the element you want them to click. For example, if there is a New button and a New link on the same screen, you might want to make it extra clear that you're talking about the button:

Next to the Foo field, click the **New** button.

Likewise, if they're on a crowded screen and you want them to click the Admin tab, you can clarify that as follows:

Click the **Admin** tab, and then click **New**.

When referring to the section using the section label itself, use the same capitalization that is used in the UI. For example:

"In the Pricing and Discounts section, do this...."

You can also refer to the section without using the actual label. In this case, do not use the UI capitalization. For example:

"Review the pricing and discount information."

Variables

When referencing a variable on the command line or in a cbean, use the character tags EI_EmphasisItalic or CV_CodeVarible. No brackets are needed. If an explanation of the variable is needed, put it on a separate line. For example:

Set to:

`(creator:/com/merced/cube/rolap/CBeanCubeComponentAdapter.`*CubeName*`)`

where *CubeName* is the name of the cube.

3 Standards Reference

- Writing tooltips, inline text, and error messages
- Sections for any chapters that have specific rules

4 Using Graphics

Question/Issue: TBD: Document FM11's object styles feature.

This chapter contains the following sections:

General guidelines and best practices

This section contains general guidelines and best practices for creating and working with graphics.

- Take screenshots in a supported client browser, such as Internet Explorer 7, 8, or 9, or Mozilla Firefox 14 or later. For the current list of supported browsers, see the *Overview Guide*.
- Use Snagit to take screenshots, and save them in PNG format.
- Import graphics by reference. This not only reduces the file size, but also makes modifying graphics much easier than if the graphics are copied into the document.
- Make sure the dpi setting you choose for new and revised graphics creates consistently sized graphics compared to others in the document. Try to stick with one or two dpi values depending on the type and size of graphics you import.
- Make sure the imported graphics are readable.
- Use FrameMaker's graphics tools to create elements such as callout text, callout rules, and borders. This makes updating graphics much easier and improves their appearance.
- Make sure the screenshots show as much of the user interface as needed. You can crop out any unnecessary areas in a screenshot to save space. Use a tool such as PhotoShop or Illustrator for this task. Don't crop the graphic by resizing the anchored frame.
- Grouping the contents of anchored frames can prevent inadvertently moving a graphic element.
- If you need to modify the text in a screenshot, use the Verdana font and make the text size 11 pixels.
- Sometimes you might need to extend the anchored frame into the margin for a wide screenshot. Use your best judgment in handling this type of graphic, and consider cropping some of the screenshot.
- Avoid adding text to screenshots that repeats what is previously stated in the body text.
- Try to keep NICE or Yosemite-specific information out of screenshots for branding purposes. For example, crop screenshots so that they do not include the NICE logo.
- You typically conditionalize anchored frames so that they do not appear in the online Help, though there might be some exceptions.

Setting up your display

Question/Issue: Also applies to Captivate?

This section describes how to set up Windows before you start taking screenshots. (This section might also apply to taking videos in Captivate.)

1. From the Windows **Start** menu, choose **Control Panel**.
2. Click **Appearance and Personalization > Personalization**.
3. Click the Desktop Background icon and choose white, and then click **Save changes**.
4. Click the Window Color icon and choose frost, and then click **Save changes**.
5. Close the control panel.

Capturing and importing screenshots

This section describes how to include screenshots in your FrameMaker files. The procedures assume you have Adobe Photoshop or Illustrator installed on you computer, though other graphics programs are allowed.

NOTE: Be sure to import screenshots into FrameMaker documents by reference.

➢ To create an import a screenshot:

1. Start Snagit.
2. Display the application screen you want to capture.
3. Select a basic capture profile and click **Capture**.

   To use additional capture options, such as taking a delayed shot or including the pointer, click an icon under **Options** and set the option before you click **Capture**.
4. Save the image under a descriptive name in the graphics folder of the guide. For example: `action_link_edit_db.png` and `schema_info.png`.
5. In FrameMaker, open the file you want to import the graphic into.
6. Inset a new paragraph below the paragraph that introduces the screenshot.
7. Tag the new paragraph FR0_Frame, FRS_FrameSideHead, or FRW_FrameWide. For a description of these formats, see Appendix A, “Using the Templates.”

8. Choose **Special > Anchored Frame**, and create an anchored frame with these specifications:

Tag	Position	Width
FR0_Frame	At Insertion Point	5.25 in.
FRS_FrameSideHead	At Insertion Point	1.25 in.
FRW_FrameWide	At Insertion Point	6.5 in.

9. Select the anchored frame and choose **File > Import > File**.
10. Navigate to the graphic file, make sure **Import By Reference** is selected, and then click **Import**.
11. In the Imported Graphic Scaling dialog box, click **150 dpi** and then click **Set**.
12. (Optional) To adjust the dpi setting, right-click the graphic and choose **Object Properties**. Click **Set dpi**, change the value as appropriate, click **Set**, and then click **Apply**.
13. Align the graphic to the top of the anchored frame and to the left margin of the preceding paragraph, as follows:
    - Under body paragraphs, right-click the graphic and choose **Object Properties**. Set the left offset to 0 and click **Apply**.
    - Under numbered steps, right-click the graphic and choose **Object Properties**. Set the left offset to 0.25 in and click **Apply**.
    - Under lettered steps, right-click the graphic and choose **Object Properties**. Set the left offset to 0.5 in and click **Apply**.
    - In table cells, create an anchored frame positioned **At Top of Column**, then import the graphic and align it top and left, if possible.
    - Look at the screenshots in this chapter for examples of proper alignment. Make sure the graphic is vertically snug in its frame to ensure proper spacing above and below the anchored frame.
    - If your graphic is too wide, you might need to align it to the right margin.
14. Select the paragraph containing the anchored frame and apply the NotInHelp conditional text setting.

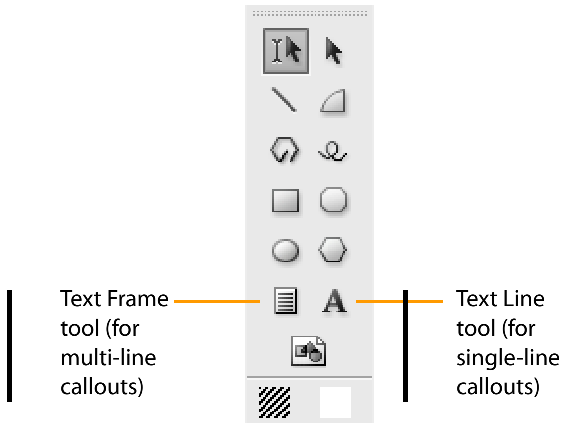
Adding callouts to screenshots

For best results, always use FrameMaker's graphic tools to create callouts.

➢ To add callouts to an imported graphic in FrameMaker:

1. For callout rules in anchored frames, use a 1.0 pt. orange line.

2. For single-line callout text, use the text line tool:
   - a. Select the Text Line tool.



   - b. Click inside the anchored frame and type the callout text.
   - c. Select the text and tag it using the FIC_FigureCallout character tag.
   - d. Select the contents of the anchored frame and choose **Graphics > Group**.
3. For multi-line callouts, use the Text Frame tool:
   - a. Select the Text Frame tool.
   - b. Draw the text frame in the anchored frame, and click **Set** in the **Create New Text Frame** dialog box.
   - c. Place the insertion point inside the text frame and tag the paragraph FIC_FigureCallout. Do not use a character tag.
   - d. Enter the callout text.
   - e. Position the graphic elements.
   - f. Select the contents of the anchored frame and choose **Graphics > Group**.
4. To emphasize areas in a screenshot, use the rectangle tool:
   - a. Select the Rectangle tool.
   - b. Choose a line width of 1.0 pt.
   - c. Choose Orange from the Set Color menu.
   - d. Draw the rectangle.
5. To create a solid border around a screenshot, use the rectangle tool:
   - a. Select the Rectangle tool.
   - b. Choose a line width of 1.0 pt.
   - c. Choose Orange from the Set Color menu.
   - d. Draw the border.

Creating and importing Visio graphics

This section describes how to include Visio graphics in your documents.

➢ To create and import a Visio graphic:

1. Use Visio to create the graphic and save it as a Visio file (with a `.vsd` file extension).
2. From the **File** menu, choose **Save As**.
3. In the Save As dialog box, choose **Enhanced Metafile (*.emf)** from the **Save as type** list.
4. Click **Save**.

   Visio converts the graphic to an EMF file.
5. Import the EMF file into your FrameMaker document as described in “Capturing and importing screenshots” on page 51.

   You usually do not need to specify a resolution as the file is imported at the actual size. Flowcharts, though, typically do need to be resized.
6. Align the graphic to the left margin of the paragraph that appears above it. Graphic alignment is the same as screenshot alignment. Look at the screenshots in this chapter for examples of screenshot alignment.

➢ To add callouts to an existing Visio image:

Question/Issue: Need to check these steps; select font.

1. In Visio, create a new blank document.
2. Choose **Insert > Picture > From File**, and then specify the graphic file that you want to add callouts to.
3. Make sure the zoom setting is 100%.
4. Click the Text tool, and then draw a text box about the same size as the callout you want to add.
5. Click the Line tool, and then draw a line from the image to the callout text.

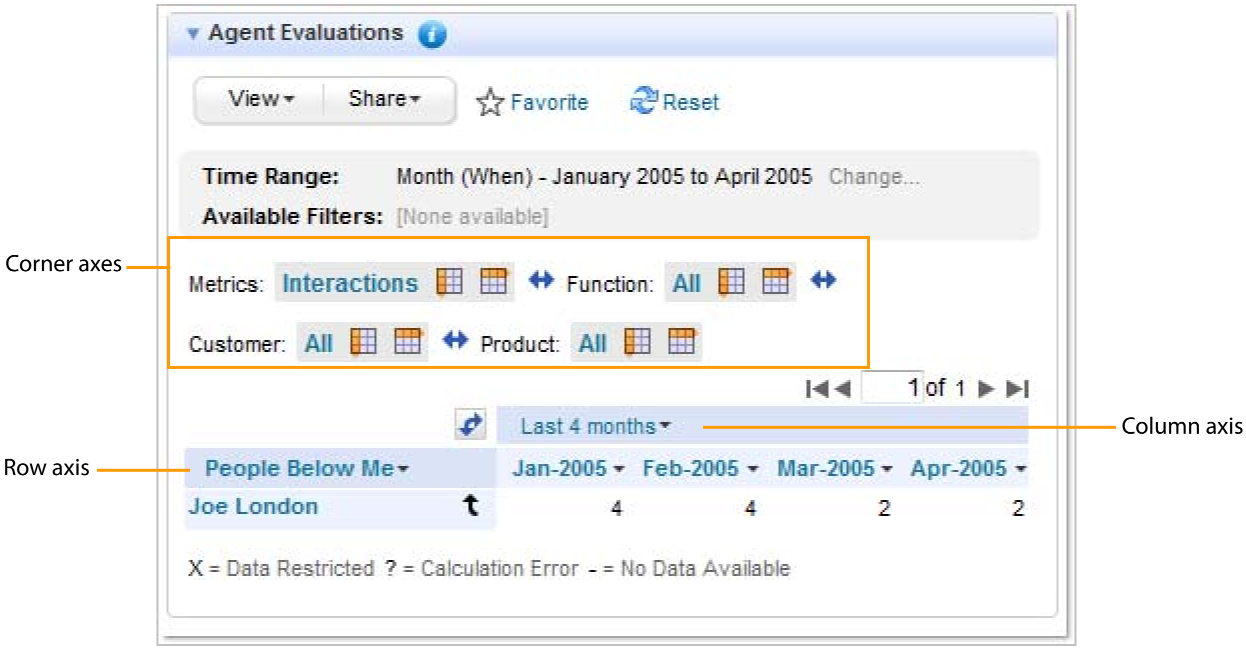
   Press and hold down the SHIFT key to create a straight line.
6. Choose **File > Save As**, and then save the file as an Enhanced Metafile (EMF).
7. Save the changes to the VSD file.
8. Follow the instructions for importing the file into the FrameMaker document.

Adding Alt text to graphics

We use graphics sparingly in the online help, so most graphics in the FrameMaker books are tagged with the NotInHelp conditional text tag. If a graphic will appear in the help, however, you must add "alt" text to it so that it can be read correctly by screen readers for visually impaired users.

For example, the NPM User Guide includes the following graphic showing the parts of a report. This is an important graphic for orienting users to the axes in reports, so we include it in the help.

Question/Issue: Following graphic imported at 96 dpi (150 specified earlier in this chapter). Does it need to be this big for RH to convert it correctly?



To allow screen reader users to interpret this graphic in the help, we've added the following alt text to this graphic:

> Report showing corner axes above the table, the column axis running along the top of the table, and the row axis running down the left side of the table.

The following procedures describe how to create alt text for graphics in FrameMaker and RoboHelp.

➢ To add alt text to a graphic in FrameMaker:

1. Right-click the graphic's anchored frame and choose **Object Properties**.
2. In the Object Properties dialog box, click **Object Attributes**.
3. In the Alternate text box, enter brief text that describes the image.

➢ To add alt text to a graphic in RoboHelp:

1. Click the graphic, and then switch from Design view to HTML view by clicking HTML at the top of the tab.

   The code for the graphic is highlighted in the HTML view.

2. Locate `alt=""` and enter brief text between the quotes that describes the image.

Working with graphics in HTML files

Question/Issue: Discuss resizing converted graphics to fit in help screen, safe area. Index section

Cleaning up the graphics directory

As a best practice after a release, clean up the graphics directory in your guides to remove unneeded graphics and make sure they are being imported from the correct graphics directories.

➢ To generate and alphabetize the list of imported graphics:

1. In Perforce, sync to the entire doc release directory. There's no need to check anything out of Perforce.
2. Open the book file.
3. Display all conditional text. FrameMaker does not include hidden graphics in the list of imported graphics, so you must display everything before you generate the list.
   a. Select all the files in the book and choose **View > Show/Hide Conditional Text**.
   b. Click **Show All**, and then click **Apply**.
4. Select the last file in the book.
5. Choose **Add > List Of > References**.
6. In the Set Up List of References dialog box, move Imported Graphics to the Include scroll list.
7. Click **Add**. FrameMaker asks if you want to generate the book.
8. Click **Update**. Ignore any warnings.
9. In the resulting `LOR.fm` file, choose **Special > List Of > Paragraphs (Alphabetical)**. FrameMaker warns you that the file isn't saved. Click **OK**.
10. Move Imported GraphicsLOR to the Include scroll list.

11. Click **OK**. The resulting `LORAPL.fm` file shows you the imported graphics in alphabetical order, as they appear in Perforce.

➢ To determine which graphics to delete:

1. In Windows Explorer, navigate to the graphics folder.
2. Create a subfolder for the graphics in use (you can name it `aaa_in_use` or something similar).
3. Move each of the graphics listed in the `LORAPL.fm` file to the new folder.
4. Move the remaining graphics into a new subfolder called `unused`.
5. Move the graphics out of the `aaa_in_use` folder.

   At this point, all of the unused graphics are in the `unused` folder, and all of the graphics in use are at the top level of the graphics folder.
6. Open all the files in the book and move any used graphics out of the `unused` folder as needed.

NOTE: As an alternative to this procedure, you can select the graphics to delete in Perforce and choose **Mark for Delete**. Open all the files in the book to see if any are missing. If the files open without indicating there are missing graphics, you can check in the deletion.

➢ To delete unused graphics in Perforce:

1. Synch to the latest graphics folder for the guide.
2. Check out each unused graphic for deletion.
3. Check in the deletion.

Graphic examples

This section contains the following examples:

- "Screenshot with callouts" on page 58
- "Same screenshot with different callout treatment" on page 59
- "Screenshot with callout rectangles" on page 59
- "Conditionalized screenshot s" on page 60
- "Overlapping screenshots with callouts" on page 60
- "Highlighted area without callouts" on page 61
- "Callouts in text frames" on page 61
- "Callout and UI help text" on page 62
- "Callouts for pull-down menu" on page 62
- "Highlighted areas without callouts" on page 62

Screenshot with callouts

Specs:

Graphic dpi: 150

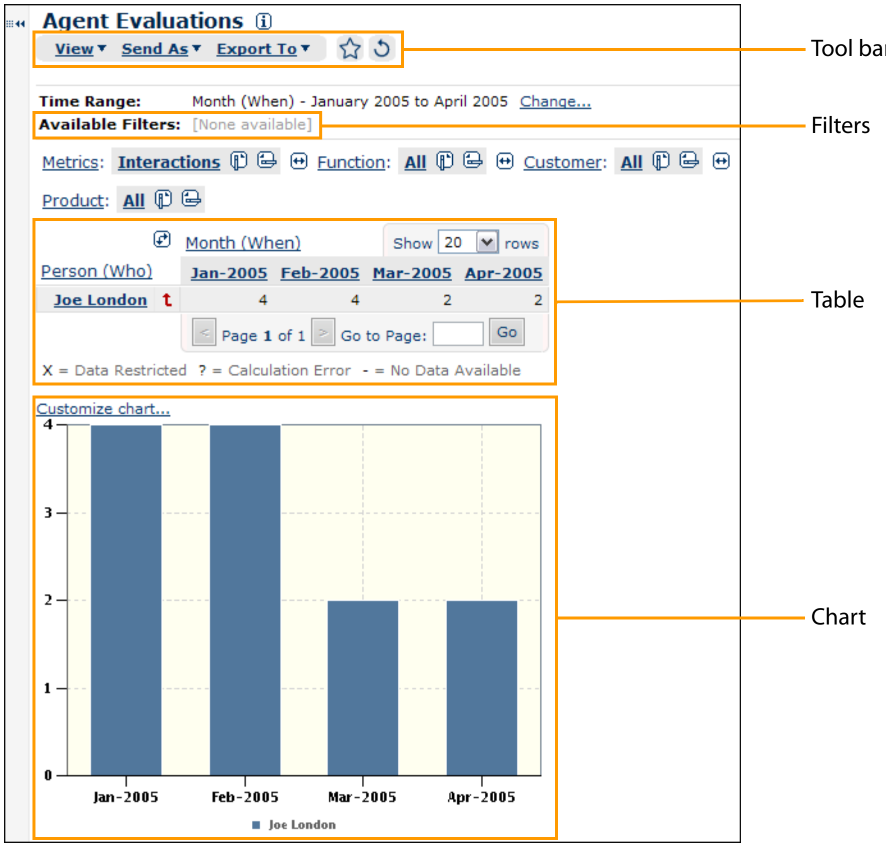
Aframe width: 5.25″

Anchored to: FR0_Frame paragraph

Anchoring position: At insertion point

Callouts: 1.0 point orange

Callout text: FIC_FigureCallout character tag



Same screenshot with different callout treatment

Specs:

Graphic dpi: 150

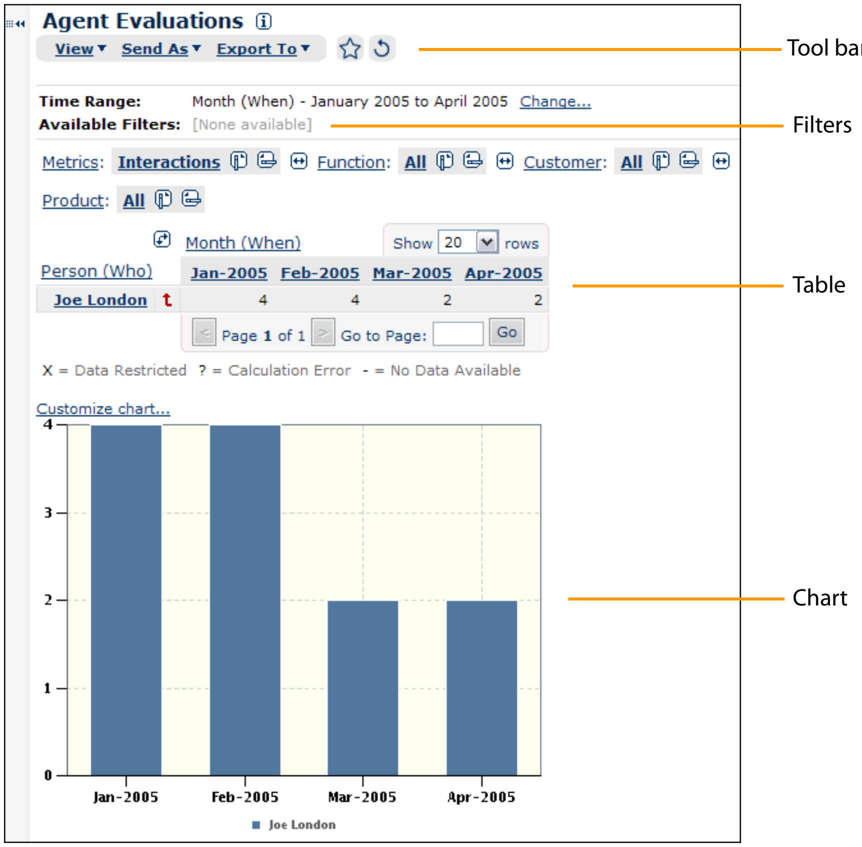
Aframe width: 5.25″

Anchored to: FR0_Frame paragraph

Anchoring position: At insertion point

Callouts: 1.0 point orange

Callout text: FIC_FigureCallout character tag



Screenshot with callout rectangles

Specs:

Graphic dpi: 120

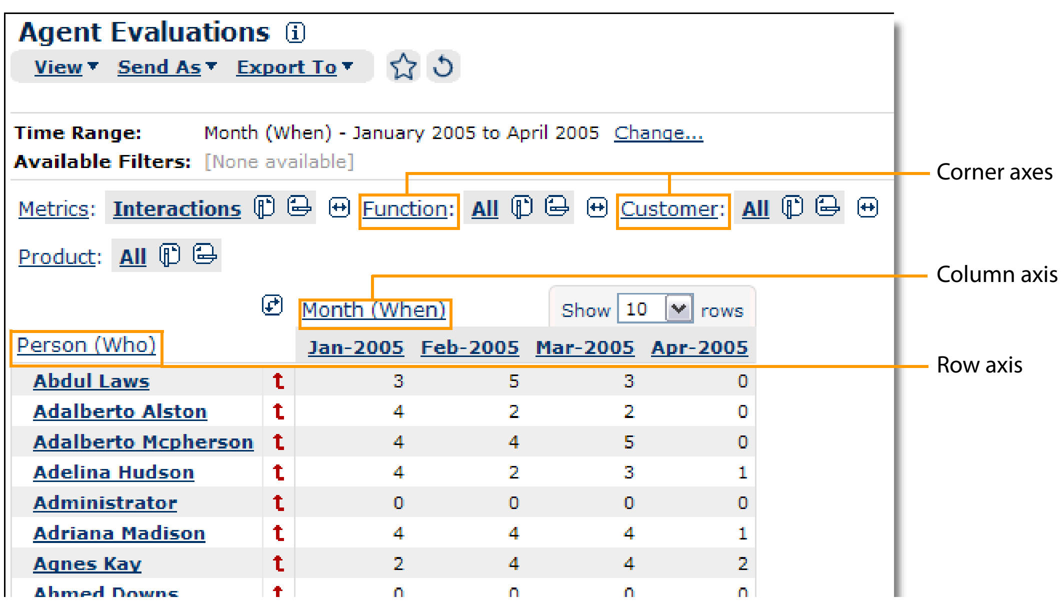
Aframe width: 5.25″

Anchored to: FR0_Frame paragraph

Anchoring position: At insertion point

Callouts: 1.0 point orange

Callout text: FIC_FigureCallout character tag



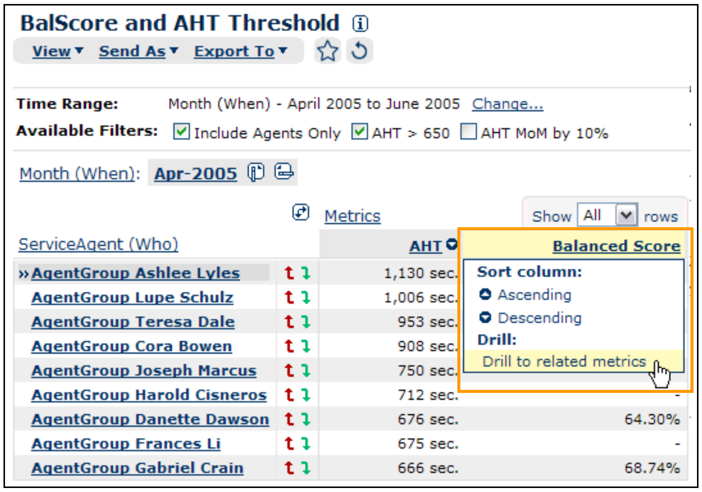
Conditionalized screenshot s

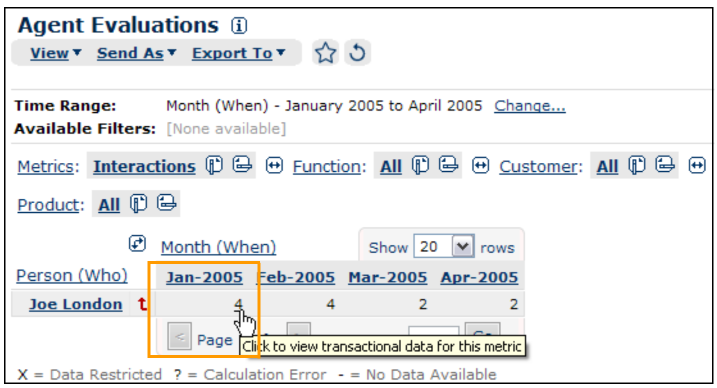
In this example, the icon in the bulleted line (but not the screenshot) appears in the online help, and the screenshot (but not the icon) appears in the PDF. To do this, tag the anchored frame in the bulleted line as HelpOnly, and tag the FR0_Frame paragraph as NotInHelp.

- While viewing a report, click **Reset**.

Overlapping screenshots with callouts

Highlighted area without callouts





Callouts in text frames

Specs:

Graphic dpi: 150

Aframe width: 5.25″

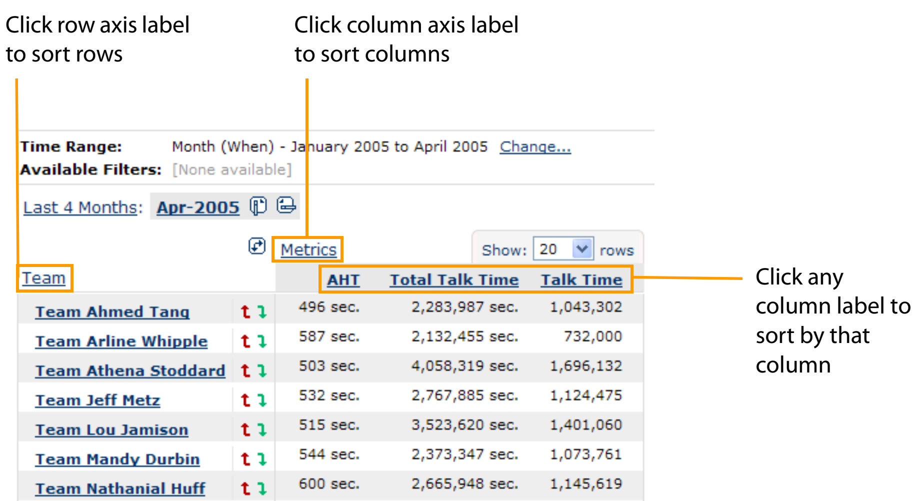
Graphic left offset: .25″

Anchored to: FR0_Frame paragraph

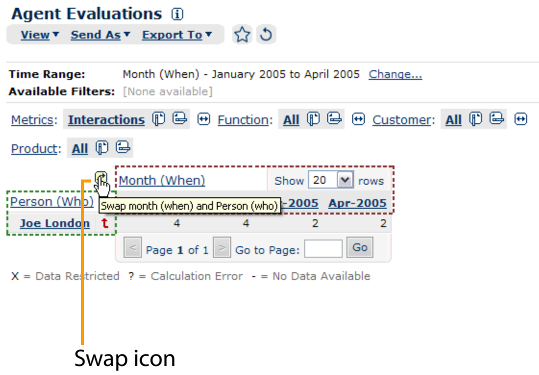
Anchoring position: At insertion point

Callouts: 1.0 point orange

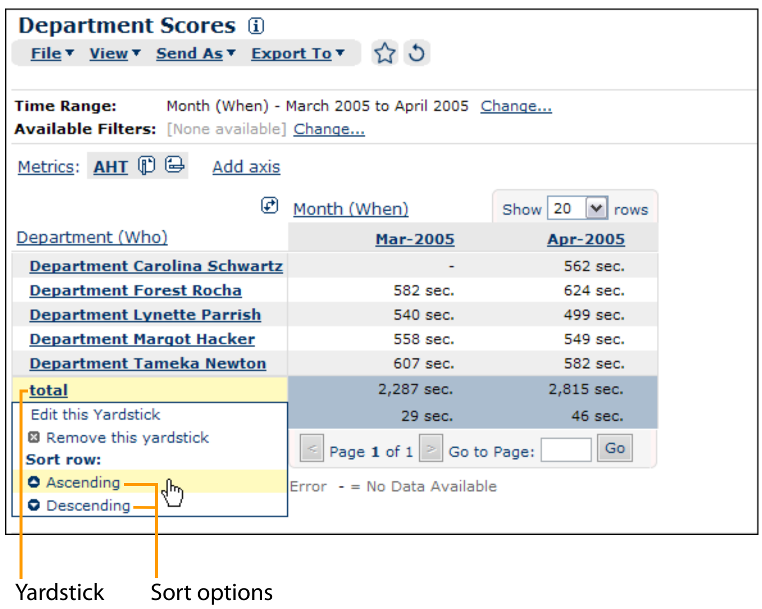
Callout text: FIC_FigureCallout paragraph tag



Callout and UI help text



Callouts for pull-down menu

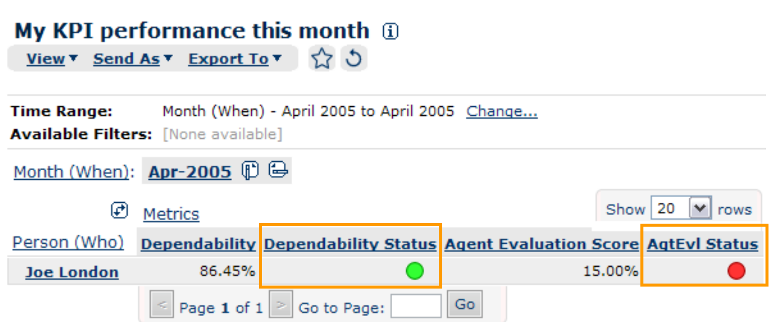


Highlighted areas without callouts

Specs:

Graphic dpi: 150

Aframe width: 5.25 in



5 Indexing

This chapter contains the following sections:

Overview

More than an expanded, alphabetical table of contents or a concordance of words and phrases, an index is a listing of the contents of a book, arranged in a predictable order, with an indication of each entry's physical place in the source document.

According to Larry S. Bonura, author of *The Art of Indexing*, "(An index) is a reader's most important map for locating information in a document that is read in a random-access style."

Regarding what is indexable, Nancy Mulvany writes in *Indexing Books* that "All information presented in the body of the text that is directly relevant to the subject matter, scope, and audience of the book is indexable."

Are indexes useful to our readers?

It's often hard to know how customers use the documentation and which tools they find most valuable. If we surveyed users, we might find that the tool most often used to locate information is the PDF search.

On the other hand, even though user surveys show a preference for text searches, usability studies show that searches have a 30% success rate, which is disappointingly low. Other studies show that using a good index is actually faster. And of course, if the user prints the document, there is no text search capability at all.

What makes a good index?

The following list describes some guidelines for strong indexes. In a good index:

- Readers can easily find the information for which they are looking.
- Every important item is indexed, but passing references are excluded or differentiated.
- Users are redirected to appropriate terms ("remove *X*" might link to a "delete *X*" topic). That is, it might contain terms that are not in the text, but for which users are likely to search.
- You provide subentries rather than strings of unanalyzed page references.
- The volume of entries is appropriate for the book. (A good rule of thumb—an index should be a minimum of 5 percent of the book's page count. For a 200-page book, provide at least 10 index pages. In most cases, closer to 10 percent is better.)

Indexes vs. search engines

Someday search engines might replace indexes for online documentation, but currently, humanly-designed print and online indexes are better at many tasks. Consider the following advantages of indexes and search engines:

Advantages of Indexes	Advantages of Search Engines
Indexes: • Have no trivial entries. The entries are those the writer has selected as substantive and relevant. • Organize entries and subentries using logical, hierarchically arranged categories. • Include terms (synonyms, implied concepts, alternate wordings) that do not actually appear in the text. • Do not require readers to type a correctly spelled, logically crafted, or grammatically correct search request. • Include pertinent cross-references to related material without diverting readers to places that offer no new information. • Are visible in entirety, providing contextual cues for readers.	Search engines are quicker at finding all instances of an exact phrase that is used in the text.

Using the IXgen tool for indexing

You can index the traditional way in FrameMaker, or you can use IXgen to help facilitate creating and editing indexes. The IXgen software is located on the I drive:

`I:\Software\Licensed Software\Ixgen`

The directory contains the software and details for IXgen installation and use. Also located in that directory are instructions for obtaining an IXgen license key. The following procedure provides detailed instructions.

➢ To install IXgen on your computer:

1. Exit FrameMaker if it is running.
2. Go to `I:\Software\Licensed Software\Ixgen`.
3. Run `setup.exe`.
4. Press Enter to start the installation, and press Enter again to verify that FrameMaker is not running.
5. Open FrameMaker.
6. Go to **IXgen > License**, and then click **Done**.
7. Send `<FM_Install>\fminit\ixgenkey.999` to Frank Stearns at `franks@fsatools.com`.

   Include your name, phone, email, and company. Within 24 hours, Frank Stearns Associates (FSA) sends you a license number.
8. Copy the license number from the email.
9. Start FrameMaker, and go to **IXgen > License**.
10. Replace the license number with the new one you copied, and then click OK.

    NOTE: Licenses are tied to computer IDs.

Here are some suggestions for more information and additional documentation:

- Download the manual ixgen7doc.pdf from http://www.fsatools.com/.
- To experiment with IXgen on safe files, use the Ecology book in `<FM_Install>\samples\overview\book`.

  Make copies of the files if you want to preserve the originals.
- Use the IXgen 7 Demonstration Guide in `<FM_Install>\ixgen\demoixgn.pdf`

For examples of using IXgen to create your index, see “Creating and Placing Index Markers” on page 124,

General guidelines for creating indexes

Although professional indexers exist, in most cases, a document’s author is the best person to index that document. Few skills can substitute for subject expertise. Putting yourself in the reader’s position and anticipating what your readers might look up to find the information they need is the key to creating strong indexes.

Keep these guidelines in mind as you index:

- Create meaningful index entries.
- Use a strong verb or noun for each primary entry. A primary entry or main entry is a first-level topic in the index hierarchy, with or without dependent secondary entries.
- Use descriptive phrases in each secondary entry. A secondary entry or subentry represents a topical subdivision of a primary entry and appears in the index indented beneath that primary entry.
- Double-post early and often. Double-posting means including the same topic in an index in at least two different locations.
- Use consistent formatting conventions.
- Provide relevant and comprehensive cross-references. Use See and See also entries for cross-references in indexes.
- Participate in indexing mini-workshops with your colleagues.

The following sections describe these guidelines in more detail.

NOTE: An implied guideline is to develop proficiency with the indexing tools available to you. For us, this is FrameMaker and IXgen. In FrameMaker, you can create or edit an index entry in the Marker window, and index entries are indicated by markers in the text. In IXgen, index entries are also referred to as markers, and you can expand or collapse the markers as you work on your index.

Creating meaningful index entries

Meaningful index entries contain information useful to your readers. For example, don’t index a term every time it’s mentioned in your book. Search engines will take a reader to each mention of a term. If you make your index entries meaningful, your readers can choose the posting that makes sense for what they want to look up. Here are suggestions for making your index entries meaningful:

- Make sure every page reference points to important information.
- Not every mention or appearance of a subject needs an index entry. For example, do not index repeated information. Index the first main appearance of a term or concept.
- Include in the index every topic treated in a major or minor heading, all special terms, and all acronyms.
- Do not merely repeat heading text unless it makes sense to do so. Reword as necessary.
- Index both features (product) and tasks (user).

  For example, index Databases:cache and Database performance:improving.
- In general, use lowercase capitalization. Copy capitalization exactly as it is in the text when it matters to do so. Make sure you enter the item exactly as written in the document (capitalization, case, number, hyphens, and so on). For example:

  ```
  Session fields
     database connection 68
     DBReader 62, 87
     DBWriter 68, 107
     RV Publisher 76
     RV Subscriber 61
  ```
- Always add identifying nouns to the names of specific parameters, functions, keys, user interface objects, commands, buttons, dialog boxes, code entities, and so forth. For example, RVNetwork parameter, StringToDate function, runtibco command.
- Pick one synonym to use throughout the index (ideally, the term that predominates in text). For example, decide upon deleting rather than removing. Then include a cross-reference to the predominant term:

  ```
  removing. See deleting
  ```
- Add synonyms when appropriate or useful. For example:

  ```
  responding to = answering
  defining = setting up = adding
  designating = assigning
  ```
- Avoid convoluted wording. Always ask, “Will the reader look this up?” For example, readers aren’t likely to look up the following:

  ```
  existing installation, adding modules to
  ```

  These entries are better:

  ```
  installation, adding modules
  modules, adding to an existing installation
  ```

Using strong nouns and verbs

Gerunds, nouns that are derived from verbs and that end in "ing," are very useful in section headings and indexes. Avoid using vague gerunds as primary entries—for example, entering or selecting. Consider what the user will look up. You'll find more examples of weak gerunds in the sections that follow.

For primary entries, select gerunds (for example, downloading, installing, debugging) whose meaning is relatively narrow.

Unfortunately, many common English verbs have too many synonyms to make useful primary entries. For example, cutting, editing, deleting, and removing are all versions of the same action, and each might apply to a wide variety of objects in your text. In such cases, including all the synonyms in the index isn't always practical. Choose the synonym whose meaning is most specific to your needs and apply it consistently throughout your index.

Examples of strong verbs

The following list includes examples of strong verbs. Remember these are suggestions only; you can use the verbs that make the most sense for the information you're indexing.

- adding (instead of appending, including, inserting)
- aligning (instead of justifying)
- analyzing (instead of checking, diagnosing)
- archiving
- closing (instead of ending, exiting, quitting)
- configuring
- converting
- copying
- customizing
- defining (instead of describing)
- deleting (instead of clearing, cutting)
- designing (instead of building)
- disabling (instead of deactivating, turning off)
- enabling (instead of turning on)
- excluding
- exporting
- hiding
- importing
- licensing
- modifying (instead of changing, editing, formatting)
- moving (instead of positioning)
- naming
- notifying
- opening (instead of loading, reading, retrieving)
- pasting
- previewing
- printing
- pushing
- restricting (instead of preventing)
- running (instead of executing)
- saving
- searching (instead of finding)
- sharing
- sorting
- starting (instead of launching)
- stopping (instead of aborting, canceling, undoing)
- testing
- troubleshooting

• logging	• updating
• mapping	• validating
• migrating	

Examples of weak verbs

Here are some examples of weak verbs to avoid using in your index entries.

• choosing	• obtaining
• creating (can be weak; try adding or defining instead)	• selecting
• designating	• setting
• entering	• specifying
• filling	• typing
• generating	• using
• including	• viewing
• listing	• writing
• making	

Using descriptive phrases in secondary entries

Try to word secondary entries so that the most important word comes first in the alphabetical listing (color design, for example, instead of designing in color). This practice is not always possible, but when applied, it helps readers find information quickly.

Do not arbitrarily limit the number of your secondary entries but consolidate them when appropriate.

Occasionally, adjectives are appropriate as secondary entries, but never use an adjective alone as a primary entry. For example, if you have this in your index:

```
colors
   about 10 –16
   black 11
   cross-platform 15
```

You could double-post the third item as follows. The example also shows another entry that begins with the same adjective:

```
cross-platform colors 15
cross-platform delivery guidelines 37
```

Don’t be tempted to index the two entries this way (with an adjective or adverb standing alone as a main entry):

```
cross-platform
   colors 15
   delivery guidelines 37
```

Analyzing primary entries

When analyzing a primary entry, go only one level down in the hierarchy, that is, to secondary entries. Although it might make sense to add a third level of sub-entry, it can get cumbersome and affect usability. Also, the FrameMaker index template supports only two levels. Readers don't think in terms of deep hierarchies when searching for a term in an index. The following example, with third-level entries, has too much hierarchy:

```
animals
   canines
      dogs 32
   felines
      cats 12
   reptiles
      iguana 87
```

The following are examples of the same information but with only one sub-level of hierarchy in the index. For example:

```
animals
   dogs 32
   cats 12
   iguanas 87
```

You might also add these double-postings:

`canines, dogs 32` (or just `dogs 32`)

`felines, cats 12` (or just `cats 12`)

`reptiles, iguana 87` (or just `iguana 87`)

Formatting secondary entries

Be consistent in the creation of secondary entries. Include at least two secondary entries beneath a primary entry. A *See also* cross-reference can be counted as a subentry, so it's possible to have a primary entry with a *See also* cross-reference and one or more subentries below it. For example:

```
SAP
   See also HRMS
   integration channel requirements 108
```

A primary entry should not have a single secondary entry. If you have a descriptive phrase for a primary entry, format it like this:

`adapters, loading 53`

or

`character encoding in CIF 30` (Note that the descriptive phrase is not always set off by a comma.)

To create a secondary entry in FrameMaker, use this syntax:

term1:*term2*

For example:

`strings:about` and `strings:parsing`

These entries might appear in the index as:

```
strings
   about 72
   parsing 78
```

Double-posting

Double-posting, sometimes called cross-posting, means including the same topic in an index in several different locations. You can do this by:

- Slightly varying the order or wording of a primary entry (reconciliation, manual and manual reconciliation)
- Including the topic as both a primary entry and a secondary entry for related topics

Double-posting offers readers access to information in the form and location that seems natural to them.

Make sure to double-post any variations on a primary entry you think readers are likely to look up (system clocks and clocks, system). Make sure the same locators (that is, page numbers) appear with all variations of the term. If the term system clocks has secondary entries, however, use a *See* cross-reference to save space. For example, clocks, system. *`See`* `system clocks`.

Double-post acronyms unless the term is one your audience is likely to know only as an acronym (a case might be HTML). Whether you double-post or not, spell out the term with the acronym in parenthesis. For example,

```
Simple Mail Transfer Protocol (SMTP) 15
```

and

```
SMTP. See Simple Mail Transfer Protocol 15
```

NOTE: If an acronym is in a secondary entry, it's fine not to spell it out if the acronym is spelled out in a

primary entry. For example:

If you have this primary entry:

```
American National Standards Institute (ANSI) 30
```

This is an acceptable double-posting (assuming there are other subentries for the primary entry):

```
units of measure
```

```
ANSI 30
```

Using consistent formatting conventions

Create index entries in the following format: primary:secondary. Very specific entries, like:

`AdjustmentTypePull` integration event

might not follow this format.

- Use lowercase text, except if the term itself is capitalized, for example:

```
log files 11, 39
Enterprise Manager 27
```

- If possible, make noun entries plural, especially when the entry refers to a generic item or has subentries, for example:

```
adapters
   about 14
   loading 16

events 56, 78

interfaces
   implementing 77
   installing 62

zones 33
```

- Use gerunds for actions, for example:

```
overriding methods 31

initializing
   classes 4
   instances 68

interfaces
   implementing 73
   installing 62
```

Avoid orphan entries. Use the following format when you have only a single secondary entry:

```
adapters, loading 15
```

Instead of:

```
adapters
   loading 15
```

Do not use locators in primary entries that have secondary entries. For the main discussion of a concept or new term, use the secondary entry about with a page locator beneath the primary entry. Use about instead of defined or described. The subentry about is sorted at or near the top of the list alphabetically. For example:

```
interfaces
   about 21
   implementing 102
   instances 150
```

Most of our guides don't contain glossaries. If yours does have a glossary, index the terms included in the glossary where they are defined and tagged as new terms in the text (and use `about` if the definition is a subentry), for example:

```
internal orders 19

permissions
   about 35
   AnalysisAuthorized 46
   BuyerAuthorized 48
   DashboardAuthorized 52
   SourcingAuthorized 70
```

Index terms in the glossary only if the terms are not also defined in the text, for example:

```
customer layer 307
```

Providing relevant and comprehensive cross-references

A *See* or *See also* cross-reference is a cross-reference in an index that appears with an entry or a subentry and has no locators of its own. Do not include locators or page numbers at the cross-referenced entry so that users become familiar with the preferred term they actually find in the text. Include pertinent cross-references to related material without ever diverting readers to places that offer no new information.

Use *See* cross-references to send readers:

- From a synonym to an equivalent term, from an industry-standard or competitive term to the NICE term, or from a less-preferred term to the one used in the documentation. (For example: `reporting grain.` *`See`* `point grain` and `branding the user interface.` *`See`* `rebranding`)
- From an overly broad topic that has no locators to more specific categories within that topic. (For example: `boxes.` *`See`* `about boxes; check boxes; dialog boxes`)
- To individual headings if the general heading includes too many items to be listed separately. Refer users to a class of terms without naming any member of the class. (For example: `commands.` *`See individual command names`*)

An entry with a *See* cross-reference cannot have anything else associated with it: no page numbers, no subentries, no other cross-references.

Use *See also* cross-references at the secondary entry level. Include the *See also* cross-reference at the top of the list of secondary entries. Use *See also* to cross-reference useful related entries. For example:

```
prices
```

```
See also tiered pricing; matrix pricing
assigning to items 16
extended 96
improving bids by 17
```

NOTE: You might rarely add a *See* cross-reference to a subentry to eliminate the need for third-level subentries.

Formatting cross-references

Use italics for the words *See* and *See also*. Begin the word *See* with a capital letter. Do not capitalize *also*.

In *See* cross-references, use a period to separate the term being cross-referenced. Do not use a period at the end of the *See* or *See also* cross-reference. For example:

```
configuration parameters. See parameters
```

Use semicolons to separate the entries in a list of terms, as follows:

```
graphics. See application graphics; button graphics
```

NOTE: To insert a semicolon so that it appears in the index, precede it with a backslash (\). For example, the format for the previous example would be:

```
<$nopage>graphics. <EI_EmphasisItalic>See<Default Para Font>
application graphics\; button graphics
```

Sort the *See also* cross-reference at the beginning of the secondary entries (without any terminal punctuation), as shown in the following example:

```
flush 3D effects
   See also application graphics
   button graphics and 82
   component spacing and 79
   default theme and 88
```

Use this format to force the cross-reference to the top of the list:

```
primary_entry:<EI_EmphasisItalic>See also<Default Para Font>
cross-reference_text[primary_entry:aaa]
```

Special indexing considerations

The following sections describe special cases to keep in mind as you index, including how to index symbols and parameters.

Symbols

If an index entry starts with a symbol such as `@` or `$`, FrameMaker sorts it under the **Symbols** heading at the beginning of the index. Double-post the entry and force the double-posted entry to be alphabetized under the first letter as well. For example, `$MERCED_HOME` would be under **M** as well as under **Symbols**.

Question/Issue: For some reason this sorting doesn't work: $MERCED_HOME[MERCED_HOME];

To force the alphabetization of an index entry, follow the entry with brackets and the spelling under which you want it alphabetized. For example, to force `$MERCED_HOME` to be alphabetized as if it were `MERCED_HOME`, you enter the index entry as follows:

```
$MERCED_HOME;MERCED_HOME
```

If you are using subentries, include the path in the brackets. For example, if the entry is:

```
tags:<HTML>
```

you would enter

```
tags:<HTML>[tags:HTML]
```

Creating an index

After you understand the concepts of what makes a good index, you can put them into practice. Creating an index consists of five phases:

- Analyzing existing entries
- Deciding what to index
- Constructing the entries (the syntax)
- Creating and placing the markers for entries
- Refining the index

The following sections describe these phases in detail.

Analyzing existing entries

➢ To improve your existing index:

1. Condense a primary entry with a single subentry into the primary entry alone. Eliminate multiple subentries if they do not add new locators to the primary entry. Reword a primary entry that has just a single subentry, for example:

   ```
   online booking
      enabling 21
   ```

   Usually the primary entry alone can do the job, for example:

```
online booking, enabling
```

2. Check the spelling, capitalization, and punctuation for conformance with the NICE documentation style. If you use the CO_Code character format for the term in text, also use the CO_Code format for the term in the index. For more information on style guidelines, see the other chapters in this guide.

   NOTE: Applying character formatting to entries is a new guideline and will take some time to implement. Use the generated index table in IXgen to apply the C-Code character format to existing entries.

3. Add appropriate nouns. If any entry needs a noun identifier, add one. For example, use PunchOut sessions instead of PunchOut.

   Whenever appropriate, change nouns to plural (in both the primary entry and subentries). For example, use notification messages instead of notification message. For more information, see "Using Consistent Formatting Conventions" on page 117.

4. Add See cross-references. Consider how the user might look up a term, even if that term is not a standard NICE term, and point the user to the NICE term. For example:

   ```
   lots. See market baskets
   ```

5. Check existing See also cross-references. Delete See also cross-references that do not offer the reader any new locators.

   If the primary entry or its subentries include any existing See also cross-references, check to see that the entries referred to appear in the index with identical wording. For example, make sure you have a matching index entry for the travel authorization orders cross-reference that follows:

   ```
   online booking
      See also travel authorization orders
      about
      Domain parameter
   ```

6. Add new See also cross-references for any related entries with locators different from the primary entry and its subentries.

   For more information on See and See also cross-references, see "Providing Relevant and Comprehensive Cross-References" on page 118.

7. Double-post variants of the primary entry. If the primary entry is a word or phrase that you think readers might look up in a slightly different form, review the index to see if those alternate forms are double-posted. If they are included, make sure all the variants have identical locators.

   Add any missing variants you think a reader might look up. (If the primary entry has subentries, however, use a See cross-reference instead of repeating all the subentries with each variant.)

   Review the subentries to see if any should be double-posted as primary entries.

   If you think a reader might look up the subentry first, double-post it.

   For example:

   ```
   parameters
      Domain
      Identity
   Domain parameter
   Identity parameter
   ```

8. Double-post any appropriate subentries as primary entries.

   For more information, see "Double-Posting" on page 116.

9. Add entries for related concepts. Add new index entries for any overarching concepts that are relevant to the primary entry and its subentries.

   For example, in a section about travel authorization orders, index the explanation of the concept of travel authorization.

   ```
   travel authorizations
   See also travel authorization orders
   about
   ```

Deciding what to index

Make sure you include every important piece of information in the index. An incomplete index is practically useless. Consider the following guidelines when creating your index:

- Use a secondary-level index to provide primary entries for categories and secondary entries for more detailed or related information.
- Index unique, well-defined concepts or details that cannot be mistaken for any other as primary entries.
- Use secondary entries to qualify, define, and amplify the primary entry. Entries should qualify, define, and amplify the primary entry.
- Limit the number of reference locators for each entry. If you have more than three page numbers for an entry, consider adding descriptive subentries for each locator or deleting the less-relevant locators.
- Avoid using repetitive words or phrases for secondary entries (for example, "how to") because they interfere with the alphabetic listing of topics and make it difficult to find specific references.

What to index as primary entries

The following tables lists items to index as primary entries.

Item	Description
Objects	Real and virtual objects ("travel authorization orders," "scheduled tasks," "notification messages") are usually the easiest index entries to identify. Include only substantive references that offer readers useful information about that specific topic. The greatest advantage of an index over a search engine is that an index includes no trivial entries.
Actions	Main tasks and procedures ("enforcing online booking compliance," "converting travel authorizations into orders," "resending failed orders") are often easy to identify, but they can sometimes be difficult to formulate into specific task names.
Concepts	Concepts ("escalations," "online booking," "visual design") are by far the most difficult index entries to identify, as they are often implicit. However, the ability to organize entries logically and conceptually is another of the great advantages that indexes have over search engines. Use that advantage.
Synonyms and industry terms	Don't forget to include common synonyms ("Data Integration and Cleansing Engine. *See* DICE") for the objects, activities, and concepts described in your text. The index is the one place in your document where you should not restrict yourself to preferred terminology. Make sure to list all industry terminology and cross-reference to the relevant NICE term.
Terms	All important and unique terminology and descriptions should be indexed.

Item	Description

Acronyms, symbols, and abbreviations	Index all acronyms. Spell out the words and put the acronym in parentheses following the term, unless the acronym is more often used. For example:
	`URL (Universal Resource Locator) 35`
	If you spell out the acronym as a primary entry, you can use just the acronym in secondary entries. For example:
	`MUSE website`
	`  platform topics on 21`
	`  URL 96`

Product features, functions, commands, and parameters	List all the main product features where they are described. Index functions, commands, and parameters when they are introduced or defined. Format the function or parameter name in the index entries as Code if the function or parameter is formatted as Code in the text.
Tables and figures	Don't forget to index substantive material in tables and figures.

Headings	Every section with a heading should have at least one primary index entry, and preferably a double-posting. So, for a section titled "Enabling the mobile application," the primary and double-posted index entries could be:
	`enabling mobile application`
	`mobile application, enabling`
	Create primary entries and double postings for all commands, forms, rules, main tasks, concepts, items, and so on.

What not to index

When you work on an index, do not index the following:

- Individual buttons, fields, or tabs unless they have a special function or a unique attribute that a user would know to look for. Be sure to index the subject of what the element does.
- Every noun. Index only important information, not passing references.
- Term definitions that are defined in the Glossary unless they are not also defined in the text.
- Notes or Warnings unless they contain information that the user might want to find and cannot find anywhere else.
- Parts of procedures except when you have procedures within procedures. (Use your judgment.)
- Titles of books. Instead, use cross-references in the text of the guide.

Constructing entries

You can use the FrameMaker Marker window or the features in IXgen to create your index entries. For more information, see "Using Consistent Formatting Conventions" on page 117 and "Using the IXgen Tool for Indexing" on page 111.

Using breakout primary entries

Set up "breakout" primary entries if the list of secondary entries is long or if there is a clear group of secondary entries. For example, the topic "reports" can be broken into multiple primary entries.

Format the breakout primary entries consistently—all commas, for example—so that they alphabetize together. The following are all primary entries:

```
active link actions
active link actions, using
active link conditions
active links
```

Creating and placing index markers

Keep the following guidelines in mind as you add index markers to your document:

- Do not spend time separating existing entries into separate markers. Instead, use IXgen to place entries in separate markers when you generate your index (select the Split concatenated index entries into separate markers option in the Apply Edited Marker List dialog box).
- Keep all markers for a topic close together, usually at the beginning or ending of a paragraph.
- Put a marker at the beginning or ending of a word, not in the middle.
- Use FrameMaker's Find/Change command to find Markers of Type:Index when searching index markers. Open the Marker window to view marker text.
- Turn on the markers with IXgen's Expand Markers command and edit them inline, if you like.
- Use IXgen to review the placement of colons and semicolons in markers.

Refining an index

Keep these tips in mind as you work on your index:

- Generate indexes (either for individual chapters or the entire book) frequently as you write, and check your work.
- Verify that every command, parameter, and so on is indexed. Are they double-posted; for example, access control groups and control groups, access?
- If an index entry looks incorrect, look for incorrect, double, or missing colons and semi-colons in the index marker or the Generated Editable Marker List in IXgen.
- Determine whether you can merge any primary entries.
- For topics with more than three page references, determine which ones point to the main information and delete any that are just passing references or repeated information.
- Run the spelling checker in generated indexes in Frame rather than in IXgen. Then you can correct the errors in IXgen more conveniently.
- Format the See and See also references correctly using the N-NewTerm character tag.
- When you finish, look for primary entries with only one secondary entry and delete the secondary entry or reword them to include the concept.
- Search the index for broken page ranges.

During production, keep these points in mind:

- Verify that index entries are accurate by spot checking two or three entries per column of the index.
- For your very last production task: before you generate the PDF, format the index to add appropriate column and page breaks. Avoid breaking subentries across a page. You can break subentries across columns on the same page.
- Make sure to reformat your index if you regenerate your book.

Generating a chapter index

You can quickly generate an index for the chapter you are working in without taking time out to regenerate the whole book. The Chapter template include a reference page that automatically formats indexes for individual chapters and appendixes. This can be useful for checking your index marker syntax as you index. To use this feature:

1. Choose Special > Standard Index, and click Yes.

2. In the Set Up Standard Index dialog box, make sure that the Include arkers of Type scroll list contains Index.
3. Check Create Hypertext Links.
4. Click Generate.

➢ To navigate to a particular hypertext marker:

Do one of the following:

- Press CTRL and ALT and click the reference locator.
- Lock the generated index by pressing ESC+SHIFT+F l k, and then click the reference locator.

To edit marker text, choose **Special > Markers** to open the Markers window, edit the text, then click **Edit Marker**.

FrameMaker indexing tips

This section contains some tips on improving your indexes.

Ignoring articles, prepositions, and conjunctions

In this set of index entries, note the conjunction and is ignored in sorting the last subentry:

```
conditional text
   adding tags 186
   best practices for 186
   documenting settings 21–22
   and localization 47
```

The marker syntax would be as follows:

```
conditional text:and localization[conditional text:localization]
```

It is a best practice in indexing not to consider articles, prepositions, and conjunctions when alphabetizing subentries because readers will focus on the first important word of the subentry.

➢ To skip articles, prepositions, and conjunctions in your index entries:

1. Type the marker text as follows:

   ```
   conditional text:and localization
   ```

2. Include sorting information in square brackets as follows:

   ```
   conditional text:and localization[conditional text:localization]
   ```

   When the index is generated, the subentry will be sorted under the word “localization.”

Using page range markers for long topics

Readers tend to place higher value on concatenated locators because they indicate a discussions that spans pages and contains continuous information. For example:

```
configuring
   Excel files 9, 35-39
   flat files 8, 40-41
```

In this example, readers might assume that single locators point to an introduction, and the concatenated locators point to a procedure.

Question/Issue: Might want to revisit this given FM10's inability to collapse ranges that appear on the same page.

If a topic spans half a page, consider inserting page range markers at the start and end of the topic. This ensures that the length of the topic is precisely shown in the generated index if the topic doesn't fit on one page.

The following shows the marker syntax for startrange and endrange markers:

```
term<$startrange>
term<$endrange>
```

NOTE: If you need to include sorting information in startrange, endrange, or nopage markers, make sure the sorting information appears last. For example:

```
term<$startrange>[term:sort]
term<$endrange>[term:sort]
term<$nopage>[term:sort]
```

Fixing broken page ranges

You sometimes see broken page ranges in your generated index. For example:

```
commands
   formatting 44-??
   indexing 120
```

This indicates a problem with the startrange and endrange markers, such as:

- A missing startrange or endrange marker
- A marker for the same term inserted between a pair of startrange and endrange markers

To fix broken ranges, you can search the chapter for Marker of Type index using the Find/Change window, or press CTRL+ALT and click the page reference in the index to navigate to the marker.

Indexing in FrameMaker

This section contains the following subsections:

- "Inserting index markers" on page 84
- "Creating Index Markers" on page 85
- "Working with index markers" on page 86
- "Generating an Index" on page 87

Inserting index markers

Where you insert index markers in a document depends on the type of index entry you're creating. The following table describes where to insert index markers in paragraphs.

To create…	Insert an index marker…
A *See* reference	In the first paragraph in the document
A startrange marker	At the beginning of the paragraph
All other index markers (a discrete entry, endrange marker, or *See also* cross-reference)	At the end of the paragraph

NOTE: Do not insert index markers within paragraphs because they can be easily deleted by mistake.

Typing marker text

Index markers contain the text that FrameMaker extracts and sorts when you generate an index. Marker text contains building blocks for page number display and suppression, and can contain any special formatting and sorting instructions. Marker text can also contain text for more than one index entry.

The elements of the marker text comprise its marker syntax. Of course, if the syntax isn't correct, the entry in the index could be incorrectly formatted, appear in the wrong place, or display any one of several problems. You'll learn more about the rules of marker syntax later.

Creating Index Markers

This section describes how to create the following types of index entries:

- Single main entries
- Special-sort entries
- Main entry with subentries
- Multiple entries in one marker
- Page ranges
- Character formats
- See and see also cross-references
- Special characters and words
- Computer literals
- Numerals

Single main entries

Special-sort entries

Main entry with subentries

Multiple entries in one marker

Page ranges

Character formats

See and see also cross-references

Special characters and words

Computer literals

Numerals

Working with index markers

Insert all index markers at the end of paragraphs, except for See references and startrange markers. Put as many index entries for each paragraph as you need in each marker, up to a maximum of 255 characters.

If you need more than one marker in a paragraph, insert them on top of each other. Don't separate them with spaces because markers can impact line breaks. While indexing, use the Find/Change command to search for and select index markers, or use the marker pod to edit marker text without first selecting the marker.

NOTE: If you use the mouse to select a marker, you could select more than one marker but only the text of the topmost marker appears in the Marker window.

Using index shortcuts

The Index Markers Document contains markers with boilerplate text for commonly used entries. To use the document, copy the appropriate marker into your document and then edit the marker text.

The following table contains examples of index entries created with the markers in the document.

Marker Type	Marker Syntax	Index Entry
Start range	lunar calendars<$startrange>	lunar calendars 133–
End range	lunar calendars<$endrange>	lunar calendars –135
Computer voice	<C_Code>m cubepop<Default Para Font> command	`m cubepop` command 86
Main Entry See	EOF. <EI_EmphasisItalic>See<Default Para Font>end-of-file<$nopage>	EOF. *See* end-of-file
Sub Entry See	Maya:cities of. <EI_EmphasisItalic>See<Default Para Font> Chichen Itza<$nopage>	Maya art of 236–239 cities of. *See* Chichen Itza human sacrifice among 43
Main See Also	folders:accessing;folders. <EI_EmphasisItalic>See also<Default Para Font> aliasing<$nopage>	folders. *See also* aliasing accessing 50
Sub See Also	Maya:cities of;Maya:cities of <EI_EmphasisItalic>See also<Default Para Font> Chichen Itza<$nopage>	Maya cities of 35. *See also* Chichen Itza

Navigating from index entry to marker:

In the index, press and hold down the CTRL key and click the page locator.

Generating an Index

Troubleshooting

This section contains common indexing problems and recommended solutions, if available. If you find indexing problems other than those listed, contact Bob.

Problems and solutions

1. Markers are too difficult to select.

   Use the Find/Change command to search for and select markers, or the marker pod to edit markers.

2. Why do I get double question marks in my index?

   Generate a list of markers for the document or book and sort the list alphabetically. The marker text must be exactly the same for each marker, except for the startrange and endrange information.

3. When I search for marker text that I know is in the document, FrameMaker doesn't find it. Why not?

   Search for one distinctive word instead of a string of words. The marker syntax might include character formatting between words.

4. If a document has a startrange and an endrange marker on the same page, what will the page range be in the index?

   When you generate the index, FrameMaker creates a single page entry if the startrange and endrange markers are on the same page. If the markers are ever separated by a page break, FrameMaker will show the page range when the index is generated.

5. My index contains the following entry. What's wrong?

   ```
   auto-lacing shoes 37–??, 38, ??–39
   ```

   The problem is caused by a marker for the term between the startrange and endrange markers. Delete the marker on page 38 and regenerate the index.

6. If I create a multiple-entry marker that contains a startrange and discrete entries, is there any way to avoid having to enter the <$singlepage> token for each discrete entry?

   Yes. Place the discrete entries at the beginning of the marker text and the startrange entry at the end.

6 Producing PDF Guides

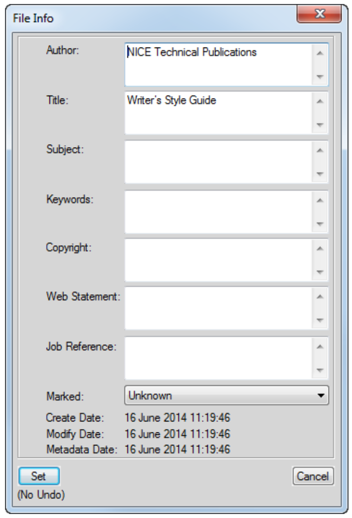
Production guidelines

Using the following checklist will help you to avoid the most common errors in final PDFs. For NICE ICM documents, use the *Final_Prod* spreadsheet in the `ICM/`*version*`/PDFs/resources` folder to track your progress.

➢ To check and generate a PDF file:

1. Sync to the entire TECHPUBS directory to make sure you have the very latest files, including templates, copyrights, shared files, and so on.
2. Check the revision date in the cover and make sure this is the same as the release date. When making emergency updates to manuals between releases, set the revision date to the current date so that CSG and others can tell the difference between the multiple manuals for the same version.

3. Right click the book in the left book pane and click **File Info**. Ensure that only the Author and Title fields are set. Author must be `NICE Technical Publications` and Title is the document title (not including the product name).



4. Search for the WC paragraph tag, `Question/Issue`, `TBD`, `XXX` and other terms that a writer might use to indicate text that should not be included or content that still needs work.

5. Open the cover file, click **Show/Hide Conditional Text** and set the **Show as per Expression field** as required. Common values for this field are:

Value	Product that the document is to be published for
Service PM Guide	NICE PM
Sales PM Guide	NICE ICM
Service PM Help	NICE PM online help
Sales PM Help	NICE ICM online help

   Select the other book files and choose **File > Import > Formats** and select **Conditional Text Settings** only in order to import the conditional expression set in the cover.

6. Check that bulleted lists are formatted correctly (for example, that the BU0_BulletedTight style is not used for lists where at least one item spans more than a single line).

7. Check that all your step intros finish in a colon (:).

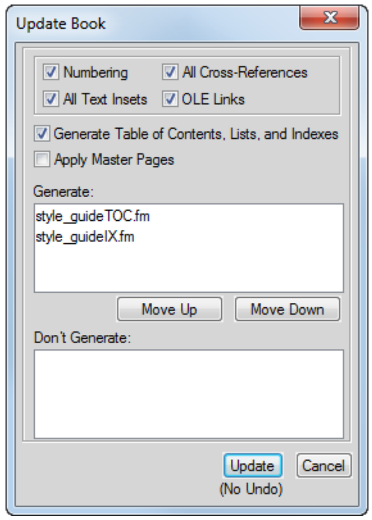
8. Check that all your screenshots and graphics exist in the book and are correctly aligned.

9. Check your index and fix errors such as broken ranges.

10. Turn off and clear change bars if you have them set in your documents.

11. Set any manual page breaks needed.

12. Run a spell check.
13. Search for any problems with conditional text.
14. Update book by selecting the book pane on the left and then choosing **Edit > Update References**. Ensure the options are set as below.

    Check the resulting book report and resolve any reported unresolved cross references or inconsistent conditional setting. You may find that some problems are falsely reported, but check them all to be sure.
15. Use the Table of Contents to check that capitalization is consistent and correct. Chapter titles should have heading capitalization, and all other headings should have sentence capitalization.
16. To generate the PDF for the complete book, select the book pane on the left and choose **File > Save As PDF**.

Generating a PDF

After choosing **File > Save As PDF**, FrameMaker prompts you for a name and location for the PDF file to be created, and also for the settings that it is to use when generating the PDF.

Filenames

When you create PDFs for individual guides, you should normally name the file as follows:

product_nameMAJOR.MINOR_title`.pdf`

For example:

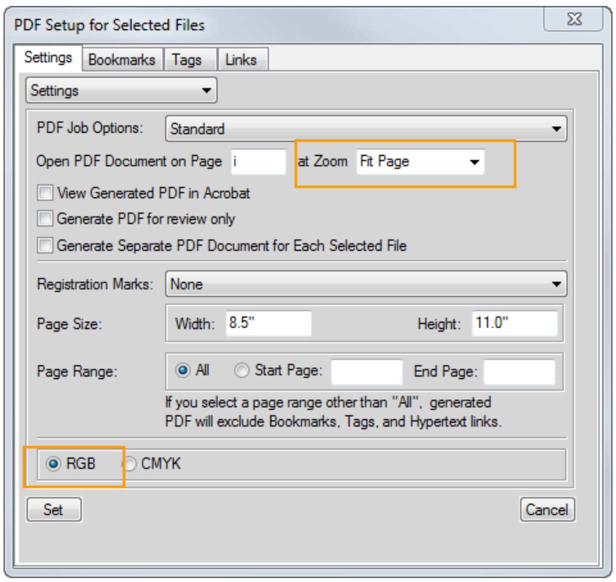
- `NICEICM6.4_OverviewGuide.pdf`
- `NICEPM6.4_AppConfig.pdf`

For most documents, do not include the maintenance release number in the filename. However, the maintenance number is included in the release guide and the portfolio filename. for example `NICEPM6.4.0_ReleaseGuide.pdf` and `NICEPM6.4.1.pdf`.

Settings tab

Check that the Settings tab of the PDF Setup for Selected Files dialog has these settings:

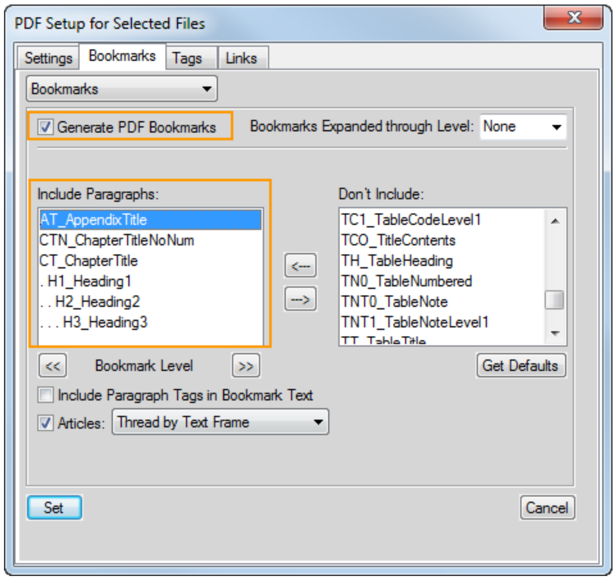
- Select **RGB** to ensure that the colors on the cover convert correctly.
- At Zoom, select **Fit Page** to set the initial zoom level in the PDF file.



The settings you choose are saved as part of the FrameMaker book file so that you normally do not need to reset them the next time you save the book as PDF.

Bookmarks tab

Make sure the settings on the Bookmarks tab include the correct heading levels (all headings to Heading 3, but not including TCO_TitleContents because the table of contents heading is not to be shown in the PDF bookmarks).



Checking your PDF files

Perform the following checks before setting the final properties and adding it to the portfolio.

- Make sure the table of contents pane expands correctly to show the headings.
- Make sure the document title appears correctly in the footers.
- Search for obsolete terminology, such as `Merced`, `Sales PM`, and `Service PM`.
- Make sure the page number shown in Acrobat is the same as the page number shown in the footer. If it is not, see if the first page numbering for the chapter immediately following the table of contents is incorrect.

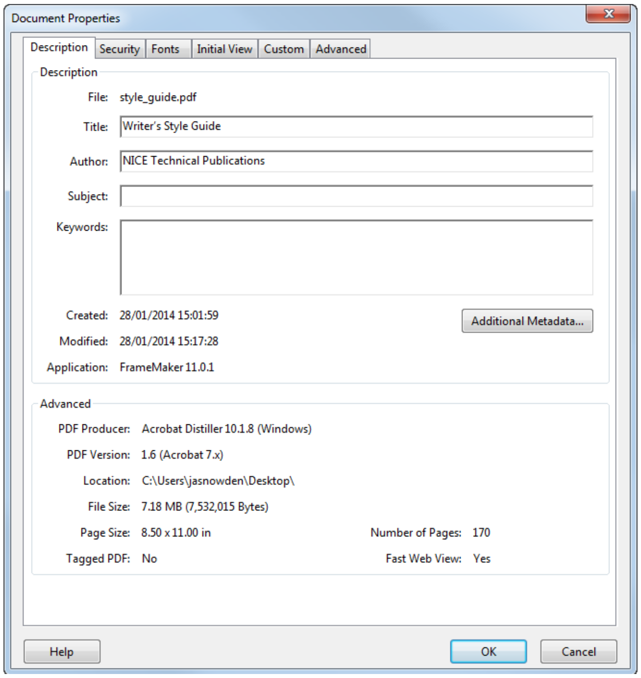
Setting PDF properties

When you are satisfied that your PDF file is correct, set the final PDF properties.

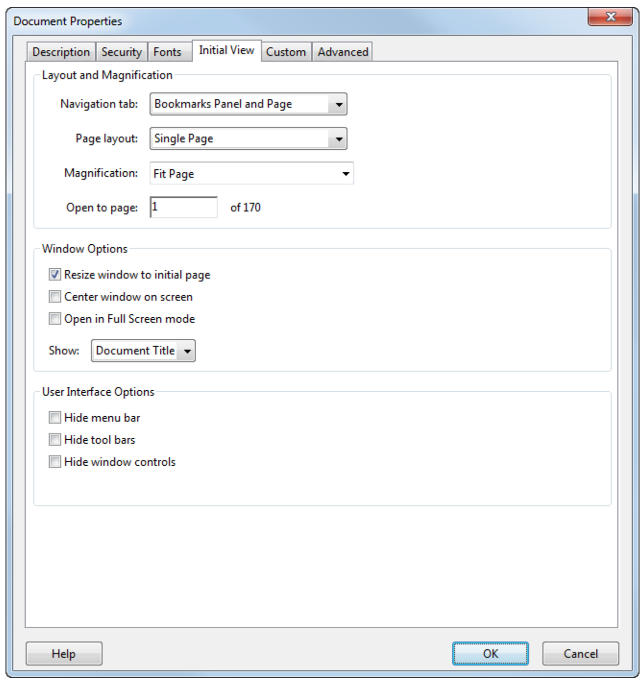
1. Click **File > Properties**.

2. On the **Description** tab:
   - Ensure the **Title** is set to the correct title for the guide.
   - Ensure **Author** to `NICE Technical Publications`.

   No other fields should be set. These values should come from the FrameMaker document if you set them at step 3 on page 90.



3. On the **Initial View** tab:
   - At Navigation tab, choose **Bookmarks Panel and Page**.
   - At Page layout, choose **Single Page**.
   - At Magnification, choose **Fit Page**.
   - Select **Resize window to initial page**.
   - At Show, select **Document Title**.



4. Click **File > Save As Other > Reduced Size PDF**
5. Set the Acrobat version compatibility to **Acrobat 7.0 and later**, and click **OK**.
6. Click **Save**.

Creating and updating a PDF portfolio

The PDF portfolio organizes all of the customer-facing documentation in one location and enables users to search across the documentation set without opening multiple PDF files. This section describes how to create a PDF portfolio.

NOTE: When preparing a portfolio for a new release, you might be tempted to reuse an existing portfolio. This can cause problems: for example, if you are using a more recent version of Acrobat Pro than the version used to create the portfolio you want to leverage. In general, it's best to create a new portfolio from scratch.

➢ To create a PDF portfolio:

1. In Perforce, sync to the latest coversheets and `Portfolio_Descs.txt` files in:
   - `//eng/TECHPUBS/SalesPM/4.1/pdfs/resources`
   - `//eng/TECHPUBS/ServicePM/4.1/pdfs/resources`
2. In Acrobat Pro, choose **File** > **New** and create a portfolio using the List with File Preview layout.
3. Add and sort the files in the correct order.
4. Add a cover sheet:
   a. In Perforce, sync to the latest coversheets in:
      - `//eng/TECHPUBS/SalesPM/4.1/pdfs/resources`
      - `//eng/TECHPUBS/ServicePM/4.1/pdfs/resources`

   b. Choose **View** > **Portfolio** > **Cover Sheet**.

   c. Click the Page Thumbnails icon. The cover page appears as a thumbnail.

   d. Right-click and choose **Replace Pages**.

   e. Navigate to the coversheet PDF.

   f. Click **Select**.
5. Add descriptions for each guide:
   a. In Perforce, sync to the latest `Portfolio_Descs.txt` file in:
      - `//eng/TECHPUBS/SalesPM/4.1/pdfs/resources`
      - `//eng/TECHPUBS/ServicePM/4.1/pdfs/resources`

   b. Copy and past the descriptions into the portfolio.
6. Choose **File** > **Save Portfolio**.

➢ To update a PDF portfolio:

1. In Perforce, copy the portfolio file and any PDFs that did not get updated to the new release folder and check in the changelist.
2. Rename the portfolio and check in the changelist.

   For example, a portfolio for a minor release might be `NICEPM4.2.1.pdf`. For information on naming portfolios, see "Checking your PDF files" on page 93.
3. Check out the portfolio and open it in Acrobat.
4. For the first updated PDF:
   a. Click **Edit**.

   b. Copy the description to the clipboard.

   c. Select the PDF to delete.

  d. Right-click and choose **Delete**.
  e. In the portfolio, select the PDF that precedes the one you are replacing.
  f. Right-click and choose **Add Files**.
  g. Select the file to add from the browser and click **Add**.
  h. Modify the display name and paste the description you copied.

5. Repeat step 4 for the remaining updated PDFs.
6. Choose **File > Save Portfolio**.
7. Click **Preview**.
8. Search the portfolio for any holes or notes you don't want to expose.

   For example, search for `TBD`, `XXX`, and `Question/Issue`.

7 Creating Videos

This chapter provides guidelines for creating and publishing videos.

The chapter contains the following topics:

General purpose of a video

The purpose of the short video clips created by Technical Publications is to give visual learners an alternative to text-only instructions. The videos are not designed to be tutorials or computer-based training.

The videos could, for example, describe:

- Common tasks that involve users having to jump all over the user interface to complete.
- Concepts that can be usefully explained with animated visuals.

The topic covered by the video must have a written equivalent elsewhere in the documentation. The videos are a visual alternative to already documented information.

Videos currently are created as SWF flash files using Adobe Captivate. Users can access videos either standalone through the customer portal or by clicking a link embedded in other media such as PDF manuals or online help.

Actions to take before creating the video

Before starting Captivate to create a video:

1. Write an audience statement that includes:
   - A description of the topic that the video covers.
   - A description of the audience the video is aimed at.
   - How presenting this topic as a video help helps the audience.
2. Write a video script covering the tasks and wording you anticipate you will use in each screen. As you actually record and develop the video, you likely will deviate from the script, but the initial script is important to provide the initial structure. It can and to enable you

Guidelines for developing the video

Follow the guidelines in this section when developing the video. This section includes the following topics:

- "Use the NICE template" on page 100
- "Use the standard screen size" on page 101
- "Use the NICE project skin" on page 101
- "Set Project Info" on page 101
- "Recommended video structure" on page 102
- "General writing guidelines" on page 103
- "Use NICE caption styles" on page 104
- "Guidelines for caption text" on page 105
- "Set the TOC" on page 105
- "Interaction screens" on page 106
- "Audio voice over" on page 106

Use the NICE template

When starting a new project, use the standard template project as the basis for the video. The template ensures the correct project size and caption styles are used.

➢ To create a new project using the template:

1. In Perforce, retrieve the latest version of the template project `captivate-style-test.cpx` from `TECHPUBS/videos/templates`.
2. Open the template project in Captivate and save as a new project in a new directory with a suitable name for the project you are creating.

Use the standard screen size

Use a Captivate project size of 720 x 540 pixels.

This size enables the video (including any sidebar TOC) to be displayed within a web browser on a standard screen size of 1024 x 768. Always use this project size for even If not using the sidebar TOC.

When recording video, use a recording capture size of 720x540 or less. When Captivate reads the captured screens into the project, it will center and pad with spaces if recording size is smaller than the project size. This can be useful if you need to add other text or objects without obscuring the capture screen.

Use the NICE project skin

Captivate uses skins to define the video appearance such as the playbar controls to use, screen margin appearance, and the sidebar TOC. Use the skin called NICE in your video projects.

To use the NICE skin in your video project, first copy the skin file to the Captivate Themes folder, and then select the skin in the Captivate skin editor.

➢ To use the NICE skin:

1. In Perforce, retrieve the latest version of the NICE skin file from `TECHPUBS/videos/templates/resources/NICE.ini`.
2. Copy the `NICE.ini` file to the Captivate `Themes` folder. This is usually located in the following location (for Captivate 5.5 under Windows 7, assuming Captivate is installed on drive C:):

   `C:\Users\`*your-username*`\AppData\Local\Adobe\Captivate 5.5\Themes`

   where *your-username* is the user name you use to log in to Windows.
3. Start Captivate. If Captivate is already running, you must close and restart it. Captivate only picks up new skins from the Themes directory when it starts.
4. Open the video project that you want to use the NICE skin.
5. Choose **Project > Skin Editor** (Shift + F11).
6. At Skin, select **NICE**. The new skin is applied immediately. There is no Apply button.

Set Project Info

You can define some general information about the project including title and copyright information. This information is not displayed to users who are viewing the published video.

➢ To set the general information:

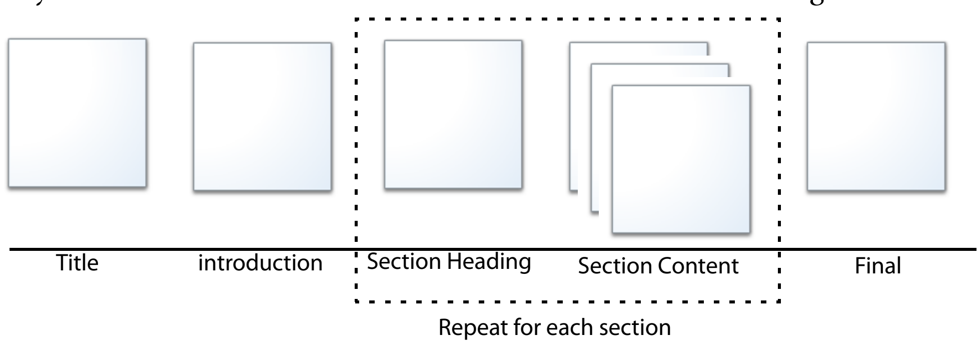
1. Open the video project in Captivate and choose **File > Project Info**.
2. Complete the project information fields as described below.

Field	Contents
Author	NICE Technical Publications
Company	NICE Systems Ltd.
E-mail	*Leave blank*
Website	www.nice.com
Copyright	Copyright © 2012, NICE Systems Ltd. Hold down the Alt key and type 0169 to get the copyright symbol.
Project Name	The video title. This should match the title on the first slide (title slide) in the video.
Description	A brief description of the purpose of the video.

3. Click **OK**.

Recommended video structure

Every video should follow a slide structure similar to the following:



Title slide

The first slide in the video should look as shown below and contain the title of the video. Include the note about no audio because some customers have been confused thinking something is wrong when they do not hear any audio.

The NICE logo is the same one as used for the FrameMaker cover for NICE PM documentation. The master is located in Perforce at `TECHPUBS/resources/graphics/other/Nice_Logo.png`.

Introduction slide

The slide following the title screen should provide an introduction and typically contain the following items:

- A brief statement on the purpose of the video
- An indication of the intended audience.
- Standard text on using the video as contained in the template. Delete the captions about using the TOC if your video does not use the sidebar TOC

Use Section Header style for the slide title and Normal text for the body text.

Section divider slide

If the video contains several sections, each section should have a section divider screen containing a single caption (style Section divider). The caption alignment should be horizontally and vertically centred on the slide.

The wording of the caption should be easy to associate with the corresponding section heading in the TOC. But because the TOC headings are so limited in text, the heading on the screen can contain more information.

Content slides

You are only limited by your imagination.

Final slide

The final slide in the video should confirm the video is ended and point where more information about the video topic can be found.

General writing guidelines

Use the following general guidelines when writing the script and creating the video:

- Assume your audience will view the complete video from beginning to end, even if you include several distinct sections in the video.
- Avoid repetition. If things are intuitive, don't repeat.
- Try to make the video as short as possible. They are really quite boring after the first couple of minutes.
- Make wording succinct to reduce the amount of on-screen reading to as little as possible.
- Follow standard style guidelines described in the rest of this guide.
- Don't refer to wording that suggests it is a training course such as *lesson*.

Use NICE caption styles

If you create a new project based on the NICE template, the correct styles are already defined in your project.

If you are editing an existing project that has not yet been converted to the NICE style, import the NICE style file to impose the correct styles. After importing the file, you may need to go through the project tagging each caption correctly. Older projects are often created by breaking all the known laws of the universe for using named styles, applying ad-hoc formatting to each caption.

The NICE style contains the following styles:

Style	Description
[Default Capture Caption Style]	Yellow Halo. Use for captions describing controls or actions on the GUI. By default, any captions created automatically during recording use this style. Modified from Captivate default.
Video Title	The title on the first and last slides.
Product Tag Line	The product series line on the first and last slides.
Section Header	Section title.
Normal text	Simple text caption. No border, transparent background. Use for blocks of explanatory text.
Section divider	Blue Halo. Large bold font. Use as first slide before a new section. Text on caption should match (but not necessarily be exactly the same) the text on the Section title.
Default Blue Hightlight Box Style	Used to highlight controls. Modified from Captivate default.

If you need to use your own custom style, try to use the following style characteristics to match the NICE style:

- Font should be Calibri, regular.
- Size for standard text is 14 point.
- Use left justification.
- If the caption needs embellishment, use one of the Captivate Halo styles as the basis.

Guidelines for caption text

Follow these guidelines when writing caption text:

- Don't apply ad-hoc styling to the whole caption. Use one of the NICE caption styles (or your own named custom style based on a NICE style).
- Punctuation. Follow normal grammatical rules, in particular ending the caption with a period if it is a complete sentence.
- Use bold to identify the following objects that would normally be tagged **GO_GUIOption** such as menu commands and field labels.
- Use `Courier New` to identify code samples/file names.
- Tip: show the background screen shot for 1-2 seconds before displaying first text caption.

Set the TOC

Use the Captivate TOC if your video is large enough to have sections that viewers might want to view selectively. To use the TOC, you must:

- Enable the TOC
- Set up the general information that is to appear above the TOC entries such as logo and title. You must set up this general information manually because it is not automatically generated.
- Select the slides that you want to appear as entries in the TOC.

➢ To enable or disable the TOC:

1. With your project open in Captivate, choose **Project > TOC** (Shift + F10) to open the TOC page of the Captivate skin editor.
2. Select the Show TOC check box.
3. At Skin, select **NICE** to ensure the NICE skin is used. The skin imposes the style settings required for the TOC including TOC position and colors. These settings are those you would set by pressing the Settings button on the TOC page of the skin editor.

➢ To set the general information for the TOC:

1. With your project open in Captivate, choose **Project > TOC** (Shift + F10) to open the TOC page of the Captivate skin editor.
2. Click **Info**.
3. At Title, type the same name that you entered for the Project Name in the general project settings.

   Leave all the other text boxes empty.

4. At Photo (under the Description box) click the folder icon, browse to the TOC version of the NICE logo, and click **Open**.

   The TOC version of the NICE logo is located in Perforce at `TECHPUBS/videos/templates/resources/Nice_Logo_TOC.png`. This version of the logo has padding above the logo to avoid Captivate running the title into the logo.

   You can preview the TOC appearance in the skin editor.

5. Close the Skin Editor (click the window close button in the top right).

Interaction screens

It is very useful to include interaction slides that require action from the user to confirm their understanding. For example, you could include a quiz or ask the user to perform a task related to the slides they have just viewed.

For NICE PM 4.1/4.1.1, these slides are not required. Further investigation of how to best include such interaction slides is needed.

Audio voice over

For releases NICE PM 4.1/4.1.1 do not use any audio voice over or background music.

For NICE PM 4.2, the Tech Pubs team will investigate text to speech using good quality TTS voices, for example those available from AT&T or Loquendo.

Publishing the video

In the Adobe Captivate software, the term *publish* describes the generation of the video files (rather than making the video files available to users). Always publish videos as Flash (SWF) files. After publishing a video, add the resulting files to Perforce.

➢ To publish a video:

1. With the project open in Captivate, choose **File > Publish** (Shift + F12).
2. Select the Flash (SWF) tab in the sidebar.
3. At **Project Title**, type the name you want for the SWF file.
4. At **Folder**, ensure the directory containing the Captivate project file for your video is selected.
5. Select the **Publish to Folder** check box. When this check box is selected, Captivate creates a subdirectory with the same name as the Project Title to contain the generated files.

6. Select the **Export to HTML** check box. When this option is selected, Captivate generates an HTML file and JavaScript file in addition to the SWF file. This HTML file can be used to preview the video in a web browser.
7. Select the **Publish to Folder** check box. This check box is disabled if the Zip Files check box is selected.

   When this check box is selected, Captivate creates a subdirectory with the same name as the Project Title to contain the generated files.
8. Ensure all the other check boxes in the Output Options section are not selected.
9. At **Flash Player Version**, select **Flash Player 9**.
10. Select the **Force re-publish all the slides** check box.
11. Click **Publish**.

When Captivate has finished generating the files, you should find a subfolder with the same name as the Project Title you typed at step 3. In this subfolder you should find three files:

- *project-title*`.swf` - This file is the main video file.
- *project-title*`.htm` - This file is a sample HTML file that can be used to watch the video in a browser.
- `standard.js` - This file contains some JavaScript functions used by the sample HTML file.

Distributing the video

You should upload all videos to the support portal. You may also decide to embed a video in a FrameMaker document so that users can view the video from within a PDF manual. If the FrameMaker document is used to generate online Help, a user can then also access the video from the online Help. You can also embed videos directly in the online help.

Uploading to the support portal

In order to publish a video to the support portal, create a zip file containing all three files output by Captivate: *project-title*`.swf`, *project-title*`.htm`, and `standard.js`. You must then upload this zip file to the support portal.

For detailed instructions, see “Publishing a portfolio on the customer portal” on page 144.

Embedding a video in a FrameMaker document

You can import the SWF video created by Captivate into a FrameMaker document. When a user opens a PDF (or online Help) that you created from the FrameMaker document, the user sees an image in place of the video. To run the video, the user clicks the image. The video plays within the PDF or online Help page by default. The user can right-click the video and choose to view it in a standalone window.

Always import the video SWF by reference into the FrameMaker document. During import you must size the video to fit within the document text column.

When you generate the PDF file or link the FrameMaker document to RoboHelp, the SWF files are embedded within the resulting PDF file and Help files.

By default, the image that appears in place of the video is the first frame of the video. You should replace this first frame with a more helpful image (termed the poster image) as described below.

➢ To import a video into a FrameMaker document:

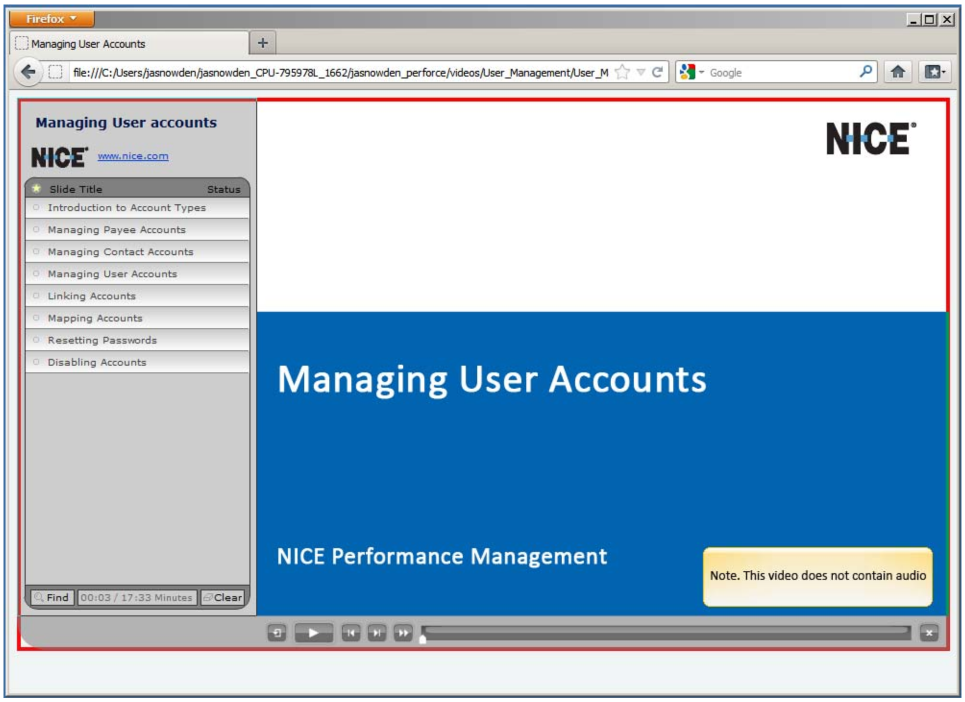
1. Retrieve the latest files from Perforce to your local Perforce workspace area.
2. Open the document in FrameMaker and at the position you want the video to appear choose **File > Import > File**.
3. Select **Import by Reference**.
4. Browse to the location within Perforce workspace that contains the `.swf` video file, select the file and then click **Import**.

   Normally, video files are located in the video project's subdirectory under the `Videos` directory in the workspace.
5. When prompted for the graphic scaling, do one of the following:
   - If the video uses the standard 720 x 540 size and uses a TOC, select a custom dpi of 190.
   - If the video uses the standard 720 x 540 size but does not use a TOC, select a custom dpi of 150.
   - If you use a non-standard video size, choose 190 dpi for the initial importing. After import has completed, you may need to amend the scaling o ensure the video fits neatly in the text area. To do this right-click the imported video in FrameMaker, click **Object Properties**, and adjust the scaling values.

➢ To create a poster image:

1. Retrieve the latest version of the video project from Perforce to your local workspace.

2. Run the video in a browser window by double clicking the `.htm` file created when you published the video. Ensure the browser window is large enough to show all the video including its controls.
3. Pause the video while the first slide is displayed, just after the note about no audio being available appears.
4. Use the capture window option of Snagit to capture just the video portion of the browser window. More the mouse over the video so that Snagit's red capture window snaps to the just video as illustrated below.



5. In the SnagIt editor, add a 2 pixel wide black border around the image.
6. Save the image as a PNG file within the video project library and ensure you add the image for uploading to Perforce.

➢ To set the poster image for an imported video in a FrameMaker document:

1. Open the document in FrameMaker and located the imported video.

2. Right-click on the video in the document, click **Set Poster**, select the poster image you created for the video, and click **Open**.

   An example of an embedded video with the poster image set is shown below.

Embedding a video directly in online Help

Question/Issue: TO BE SUPPLIED.

Tracking video status

If there is not enough time to update a video before a release, we might decide to pull it or hide it. This sort of lightweight tracking information is important to capture so that others know where a video appears in the doc set (for example, if it's embedded in a chapter), what needs to be done to update it, and when that work is scheduled to be completed.

Use the twiki page for videos and white papers to capture this information.

➢ To track video status:

1. Go to http://intranet.mercedsystems.com/twiki/bin/view/Merced/Screencasts.
2. Click **Edit** and log in.
3. Add status in the Notes column.

   For example:

```
Embedded in Sales PM help and Sys Admin guide. Pulled from 4.1.0;
added back in 4.1.1 with new branding info.
```

4. Click **Save**.

8 Creating Online Help

This chapter includes the following sections:

Overview

We use Adobe RoboHelp to generate the help systems. The NICE ICM help is sourced mainly from files created in RoboHelp, while the NICE PM help is sourced mainly from the same FrameMaker source files used to create the guides, which are converted to HTML by RoboHelp. The help files are in HTML format.

The help projects can be found in:

- (NICE ICM) `TECHPUBS\platform\`*version*`\MercedHelp`
- (NICE PM) `TECHPUBS\NPM\`*version*`\help\NICE Performance Management Help\`

To create the NICE ICM *User Guide*, we use the final HTML files to create a PDF book in Microsoft Word. However, because most of the NICE PM help is single-sourced from FrameMaker files, we have a user guide FrameMaker book from which we generate the *User Guide*.

This chapter describes the help conversion process, writing and style guidelines specific to online help, and tips for using RoboHelp.

NOTE: Useful RoboHelp fixes and workarounds can also be found at http://www.grainge.org/.

Working with RoboHelp

This section covers the following topics:

- "Introduction to the help system" on page 114
- "Setting up a new project" on page 114
- "Working with topics" on page 116. This topic explains how to create and delete topics in RoboHelp and how to ensure these changes are reflected in Perforce.
- "Indexing (ICM only)" on page 117
- "Authoring in RoboHelp" on page 118. This topic provides information on how to implement standard writing techniques such as lists and tables within RoboHelp.
- "Implementing context-sensitive help in NICE applications" on page 124

Introduction to the help system

The basic unit of the help system is the individual help topic. In RoboHelp, each help topic is an individual HTML file. The creation and deletion of files is performed by RoboHelp under the covers every time you create or delete a help topic.

NOTE: This automatic creation/deletion of files by RoboHelp can cause difficulties when checking projects back into Perforce. Advice on working with Perforce, where relevant, is given in the individual procedures below.

Tables of contents collect individual topics together into a full help system, and a table of contents can include other tables of contents.

A set of topics, tables of contents, indexes etc is called a RoboHelp *project*. Click on the project file (.xpj) to open the project. Any aspect of this project can be edited from the RoboHelp interface. Many RoboHelp settings, such as conversion settings, are saved as part of the individual help project.

Fore more detailed information on using RoboHelp, see the Adobe RoboHelp help and forums.

Setting up a new project

If your project will only use a single FrameMaker book, or no book at all, then you can create the project straightforwardly with no additional design considerations.

If your project might include multiple linked FrameMaker books, you should consider the problem of files that appear in more than one of those books. See “Sharing content between help projects with linked FrameMaker files” on page 132.

➢ To create a new project:

1. Start RoboHelp.
2. Click New Blank Project.
3. In the New Project Wizard, enter:
   - A title for the project, such as ‘Fredburger Help’. This title will appear in the RoboHelp title bar when you work with this project. The project filename and location also default from the project title, but they can be changed.
   - A file name for the project file, such as ‘fredburger’. This filename will also be used for a number of internal project files.
   - The location where all the project files will be saved. RoboHelp only uses this location for the initial creation of project files. After the `.xpj` file has been created, the project files can be moved around at will, and RoboHelp automatically creates and saves files relative to the `.xpj` file.
   - A title for the first topic. All help projects must include at least one topic file. The topic you choose to create first does not have any special status within the project

Adding standard styles

By default, a new RoboHelp project uses a stylesheet called `default.css`. You should replace this with the standard NICE stylesheet.

➢ To replace the default stylesheet:

1. In the project pane in RoboHelp, open the **Project Files** folder.
2. Right-click on `default.css` and rename it to stylesheet.css.
3. Save the project and close RoboHelp.
4. Replace your project’s stylesheet.css with the version found in source control in `TECHPUBS > resources > help`.

The next time you open your new help project, all the standard NICE styles will be available in all topics.

Working with topics

Creating topics

Before creating a topic, ensure that the topic is needed. Try to avoid proliferation of HTML files, but conversely do not overload existing files with extraneous information that merits a topic of its own.

Editing topics

When you make substantive edits to a topic, or consider restructuring a section in RoboHelp, ensure that clicking help from the application calls the appropriate topic (this is known as *context help*). Check the friendly name in the map file to ensure that topics are linked to the correct map IDs. For more information, see "Context-sensitive help in NICE ICM" on page 126.

Deleting topics

You can remove a topic from the table of contents without removing it from the project. Removing a topic from the table of contents does not remove references or links to it from other files, and that if there are such links it will still be included in the generated help and available to users through, for example, searching.

➢ To remove a topic from the table of contents:

1. In the table of contents, select the topic (make sure you select the topic in the ToC, not the file in the Project Manager pod) and click delete.
2. Save the project.

➢ To remove all references to a topic:

1. Right-click the topic and select **Show Topic References**. This opens a pop-up that shows links, index references, and other references to this topic from other files in the project.
2. To remove one of these references, select it and click **Edit**. The topic, map file or other object containing the reference will open.

   NOTE: You cannot edit index entries from the Show References pop-up. Instead, make a note of any index entries that the topic should be removed from and perform this task manually from the index pane.

3. Locate and remove the reference.
4. Repeat steps 2 and 3 as needed.

In some cases, such as when obsolete features have been removed from the code, you might want to remove a file from the project entirely. You can use the previous method to remove references before you delete the file, but it is often simpler to remove the broken links afterwards. The following procedure describes the latter approach.

➢ To delete a file from the project:

1. Select the file in the Project Manager pod (not the Table of Contents) and click delete.

WARNING: Do not click **Yes** on the pop-up that appears. If you click **Yes**, RoboHelp will remove only the href tags and leave the text of the links untouched. For example, a link to the deleted topic Add a wotsit would become the plain text 'for details, see Add a wotsit'. These plain text references are almost impossible to track down for deletion.

2. Click **No** to delete the file.
3. Make a note of the filename of each file deleted. This will be helpful when checking the project back in to Perforce.
4. In the Project Manager pod, click on the Broken Links folder.
5. Open each broken link in turn and click Edit to manually remove the reference to the deleted file.
6. In Perforce, mark all deleted files as deleted before checking the project back in (you might need to revert them before you can delete them). Remember that you need to delete the same files from the location of the help output.

Indexing (ICM only)

In RoboHelp, topics are indexed rather than individual words. To create an index entry, open the index from the project pod and add the relevant keyword or keywords. You can then drag-and-drop topics into that keyword.

For information on linking FrameMaker files into a RoboHelp project, see "Generating an index from FrameMaker index entries" on page 131.

Avoiding bookmarks in the index

RoboHelp lets you drag-and-drop topics and bookmarks into its index keywords. This works correctly in the help, but in Word output the bookmarks are not included in the index entry and break the formatting for the relevant keywords. Avoid using bookmarks in an index as far as is possible.

Copying and pasting index entries

In the index, you can create new entries using copy and paste. Copying an index heading will also copy all subheadings. This shortcut is helpful for double-posting in the index.

For example, you have a set of topics under a main index heading, and you want to duplicate these under a second heading.

1. Select the main index entry and copy (either by right-click or by CTRL-C)
2. Right-click the heading to paste it into
3. From the drop-down menu, click **Paste Inside**.

This creates a sub-heading with the name of the main index heading within the second heading, containing all the same topics.

If you try to paste the index heading where a heading of the same name already exists, the version you paste will be prefixed with "Copy of." You can then change this as appropriate.

Authoring in RoboHelp

This section covers issues relating to writing in RoboHelp, including formatting and accessibility.

Accessing the Styles and Formatting pod

To apply formatting, you use the Styles and Formatting pod that appears as a tab on the right side of the screen next to the Resource Manager. If you accidentally close the Styles and Formatting pod, you can display it again by choosing **Format > Styles** (available only when a topic is open).

NOTE: This is a departure from the convention of all other pods, which you access by choosing View > Pods and selecting the pod's name.

By default the Styles and Formatting pod only shows paragraph and character fonts. To show other styles, such as list styles, click the drop-down menu at the top of the pod and select **All available styles**.

Using Styles

Apply styles to words or paragraphs by selecting text in the topic's Design view and clicking on the style to apply.

Some styles in the standard stylesheet exist for compatibility with FrameMaker. For example, BU0_Bulleted and N0_Numbered are FrameMaker list styles that cannot be well converted to RoboHelp lists. Do not use these styles when authoring in RoboHelp, as they are optimized for ease of integration with FrameMaker rather than for appearance in HTML or ease of creating PDF output.

For example, to create a list use the **listbullet** style rather than the **BU0_Bulleted style**.

Working in the HTML view

Most formatting other than applying paragraph and word styles, such as lists and tables, will require you to edit the topic HTML directly. Click the **HTML** button to access the topic source code.

WARNING: Do not switch between HTML view and Design view while you edit. Complete all your changes and check your code before switching back into Design view. This is because when you switch back to Design mode RoboHelp automatically validates your HTML and makes any changes it considers necessary. These may not be the changes any sane application would consider necessary, and they cannot be undone.

Go back to the **HTML** view briefly after switching to **Design** view, to check that RoboHelp has not made any unwanted corrections.

Example of unwanted corrections: missing closing tags

A common mistake when working in HTML View is to delete or forget to insert a closing tag. If this happens, RoboHelp will apply that tag to everything from that point on.

For example, the following HTML has a missing span closing tag for the bold style in the first paragraph:

```
<p class="bodytext">Click the <span class="specialbold">add icon.</p>
<p class="bodytext">A new widget will be added.</p>
<p class="note"><span class="specialbold">Note:</span> Widgets are
bad.</p>
```

RoboHelp erroneously adds tags that bold subsequent paragraphs as follows:

```
<p class="bodytext">Click the <span class="specialbold">add icon.</
span></p>
<p class="bodytext"><span class="specialbold">A new widget will be
added.</span></p>
<p class="note"><span class="specialbold"><span
class="specialbold">Note:</span> Widgets are bad. </span></p>
```

There are two ways to remove the extraneous tags:

- Remove all the character formatting from the paragraphs. Go to **Design** view, select everything that you want to remove the span tags from, and click on the style **Default paragraph font**. This removes everything but the basic paragraph formatting. You might, however, need to search for and delete empty span tags.
- Preserve the character formatting in the paragraphs. Switch to **HTML** view and do a search and replace for the opening tag that you want to remove. Don't worry about the unmatched closing tags; RoboHelp will remove these automatically when you switch back to **Design** view.

Creating lists

You can quickly create simple lists in RoboHelp, but complex lists, such as multiple paragraphs within a list item or lists containing more than one level, require editing the HTML in order to get them to work properly. Avoid deeply nested multilevel lists because they are difficult to maintain and prone to errors. For example, instead of creating a deeply nested list you might split the list up into several separate lists, each with its own subheading.

➢ To create a simple list:

1. Write the text you want to appear in the bulleted list.
2. Apply the **listnumber** or **listbullet** list style as needed.
3. Select the list and click **(none)** in the Styles and Formatting pod to ensure that no paragraph styles are applied to the list items.

In previous versions of RoboHelp the **ListMapping** paragraph style was required to ensure that lists were correctly formatted when producing PDF output from the help. In RoboHelp 9 and later the use of this style is deprecated, but probably harmless.

Note that multiple paragraphs can be inserted in each list item.

➢ To create a multi-paragraph list item:

1. Click **HTML** at the top of the topic view to enter HTML mode.
2. Add the appropriate number of extra paragraphs within the `<li...> </li>` tags.
3. Use the **listcontinue** or **listnote** paragraph style for each new list paragraph.

For example:

```
<li class="listnumber">In the menu bar click on <span
class="menuoptions">Maintenance &gt; Plan Templates</span>.
<p class="listcontinue">A list of financial years for which plan
templates exist is displayed.</p></li>
```

➢ To create a multilevel list:

1. Select the list paragraphs to indent.
2. Click the Increase Indent icon.
3. Move (or cut and paste) the sublist inside an `<li...> </li>` tag, as shown here:

```
<li class="listnumber">From the <span class="specialbold">Access
Mode</span> drop-down list, choose:
   <ul class="listbullet2">
      <li class="listbullet2"><span class="specialbold"
      style="font-weight: bold;">Manual</span> to run the data port
      manually, or chained by another data port.
      </li>
      <li class="listbullet2"><p style="text-indent: 0px;"><span
      class="specialbold" style="font-weight: bold;">Auto Run</span>
      to run the data port automatically. Authorized users will still
      be able to run this data port manually if required.</p>
      </li>
   </ul>
   <p class="bodytext">If you set the report to run automatically, you
    also need to set when it runs.  See <a target="_self"
   href="7662.htm">Auto Run Data Port Options</a>.</p>
</li>
```

This example appears in the **Design** view as:

```
1. From the Access Mode drop-down list, choose:

   •  Manual to run the data port manually, or chained by another data
      port.

   •  Auto Run to run the data port automatically. Authorized users
      will still be able to run this data port manually if required.

   If you set the report to run automatically, you also need to set
   when it runs.  See Auto Run Data Port Options.
```

NOTE: Nesting sublists within individual items is the approach recommended by W3C, and it is how multilevel lists are implemented throughout the help. Any other approach might not appear correctly in all output formats.

Inserting images

When deciding which images to use in the help, remember that the help is often called through the UI, so screenshots are usually not helpful. Conceptual graphics that pertain to documented tasks may be useful, though.

Click **Insert** > **Image** to insert an image from anywhere on your computer.The image will automatically be copied into the project root folder.

Remember to add the new project file into Perforce, in both the help source and the help output locations.

Accessibility

All images need useful alt text to meet Section 508 requirements for software applications. Screen reader users hear the alt text descriptions that are the attributes of the graphics.

Alt text such as “A screenshot of a report” is not useful because it does not provide the same information as the image, such as how the information is laid out, and what icons are present and where they appear. An example of the alt text describing a thermometer-style report output is as follows:

```
<img src="images/widget_thermometer.gif" alt="An example of this
widget looks like an old-fashioned glass mercury or alcohol
thermometer. Numbers up the side range from 0 to 250. To the right, a
key shows that the red-colored ‘mercury’ within the thermometer
represents the number of units sold."
```

Inserting tables

➢ To insert a table:

1. Click **Table** > **Insert** > **Table...**
2. Set the appropriate number of rows and columns (remember to count header rows).
3. In the **Style** area, select **tabledefault**.
4. Enter text in the table. Use the special **tablebodytext** and **tableheading** settings for the main text in the body and heading cells. You can use **specialbold**, **note** and other standard styles for formatting.

Accessibility

Table headers

Make sure that all table headers use the <th> tag rather than the <td> tag, and use the scope=”col” or scope=”row” option to specify whether they are a column or a row header. For example:

```
<table>
   <tr>
      <th scope=”col”><p class=”tableheading”>Command</p></th>
      <th scope=”col”><p class=”tableheading”>What it does</p></th>
   </tr>
```

Many screen readers read tables along from left to right and top to bottom. This can make it difficult for users with screen readers to orient themselves within a table. Using column and/or row headings is very helpful for users with screen readers that can read out the headings associated with each cell.

Tables imported from FrameMaker topics do not have the scope tag, so you must update them during production (see “Generating accessible help” on page 136).

Complex tables

Try to avoid tables with merged cells or subheadings, but sometimes they are necessary. For complex tables, use the `summary` attribute to describe the table layout.

```
<table summary=”Table of commands. The table rows are grouped into
sections, each of which has a separate section heading on the row
above.”>
```

Where a table has cells spanning multiple rows or columns, or subheadings, screenreaders may be unable to tell which headings are associated with which body cells. You can specify the headers for any specific cell using the `headers` and `id` tags, as shown here:

```
<table>
    <tr>
        <th id=”func”><p class=”tableheading”>Function</p></th>
        <th id=”action”><p class=”tableheading”>What it does</p></th>
    </tr>
    <tr>
        <th colspan=”2” id=”maths”><p class=”tableheading”>Mathematical
        functions</p></th>
    </tr>
    <tr>
        <td headers=”maths func”><p class=”tablebodytext”>ADD</p></td>
        <td headers=”maths action”><p class=”tablebodytext”>Adds two or
        more numbers together.</td>
    </tr>
```

NOTE: A table cell can have more than two headers, but if it has more than three your table is too complicated and will be difficult for users to navigate and for you to maintain. You should try to restructure it or break it into multiple separate tables.

Hyperlink properties

Every time you add a new hyperlink, set the hyperlink properties to Display in frame: Same Frame. If you don’t do this, every link the user clicks on will create a whole new window. If you find a way to set the entire project to do this by default, please describe it here.

Showing the table of contents in output

By default, when the user opens the help and a context-sensitive topic is triggered, that topic appears in a single pane with the TOC and other controls hidden but with a “Show” link at the top of the window. Since “Show” isn’t very informative, you should change this to “Show Table of Contents.”

➢ Changing the “Show” link in RoboHelp:

1. Choose **File > Project Settings** and click the General tab.

2. Click Advanced, and then click the LNG File tab.
3. Scroll down to the [Common] section and change `Show=Show` to `Show=Show Table of Contents`.

Using user defined variables

User Defined Variables allow you to create reusable strings of text within your help topics. These work in a similar way to FrameMaker variables.

NOTE: RoboHelp automatically creates user defined variables for all the variables in any linked FrameMaker files. This results in some proliferation of variables.

1. You can create a new variable either from the User Defined Variables pod or by right-clicking and selecting **Add To > User Defined Variable**.
   - Service PM: *MAIN*`/mps/products/cc-enterprise/src/Urls.properties`

Reports and searches

RoboHelp provides a number of reports to enable you to find out information about the project such as which files are unused. You will probably only use these rarely, but they can be a powerful and useful tool. All reports can be found under **Tools** > **Reports**.

NOTE: The Unreferenced topics report only checks the default ToC. Topics that are used in other ToCs but not in the default ToC will show up as unreferenced.

Using find and replace

If you need to find and replace a term or a snippet of HTML throughout a set of RoboHelp files, use a separate find and replace application such as FAR (for which we have two licenses). RoboHelp's own Find and Replace in Files option can be useful for simple searches, but has historically not been terribly good.

Implementing context-sensitive help in NICE applications

This section describes how to implement a context-sensitive help system in the NICE NICE PM and NICE NICE ICM applications.

Context-sensitive help in NICE PM

To call a specific help topic when viewing a specific screen, the cbean for that screen must contain a doc-key property (docKey also works) that calls an ID number (such as 11001) that maps to the correct help topic in RoboHelp. As a new feature is being developed, you should give the developer the ID to use in the doc-key property and then ensure that it's properly mapped in the help system. You can also modify the cbean file yourself, but use great caution, because many text editors will mangle the Asian characters in the files and/or insert odd characters at the beginning of the file that break the build. Notepad++ is the only text editor I've found that works correctly on Windows.

If you can't get a developer to implement the doc key for you, or you need to troubleshoot a doc key, take the following steps:

1. In the application, right-click the page, choose **View Page Source**, search for `OpenHelp` to find the `mercedOpenHelp` function, and note the ID that's being called (the first parameter). For example, in the configuration manager, when you view the page source and search for `OpenHelp`, you see that the ID is 93:

   ```
   'action' :
   'mercedOpenHelp("93","\u002fmerced\u002fdocumentation\u002f\u002fc
   om\u002fmerced\u002fdocumentation\u002fServicePMHelp\u002findex\u0
   02ehtm")'
       });
   ```

   If the first parameter is blank or is a string, you must change it to a numeric ID in the cbean. The next steps will help you find the correct cbean.

2. In Perforce, click the Find File tab in the right pane, set the Name contains field to ".cbeans" to filter on the cbeans files, and click **Find**. This finds all cbean files in the depot.

3. Select all the cbean files and sync to the latest revision. Now you have all the cbean files in your workspace and on your local disk, which will allow you to search through them using Notepad++.

4. Open Notepad++, choose **Search > Find in Files**, enter the ID that was on the page, and for the directory specify the root of the release branch you're working in (such as `C:\p4\jwheeler\unified\releases\NICEPM\41MAINT` for the 4.1 maintenance branch on Jackie's machine).

   If there wasn't an ID, enter the end of the page's URL, which will sometimes take you to the correct cbean. For example, when you go to the New Record screen in Service PM, the end of the URL is HierarchyAddRecordContentItem. If that doesn't help, you can also use the cbean navigator to navigate to the cbean using the page's URL starting from "merced" (for example, `merced/content/admin/left-nav/hierarchymanager/HierarchyAddRecordContentItem`). When you get to the right cbean, you can search in Notepad++ on the text you find in that cbean to get to the right file.

5. When you identify the correct cbean file, add the doc-key property if it doesn't already exist, or update the existing one with the correct ID. If you add the property, be very careful that you're honoring the structure of open and closed parentheses. For best results, look for another doc-key property to use as an example.

6. Make sure the ID you use is in the mapping file in RoboHelp and that it doesn't conflict with any IDs in the other product's help system.

Context-sensitive help in NICE ICM

RoboHelp uses *map files* to create context-sensitive help. These map files link friendly names (such as HELP_TRANSACTIONS) and map numbers (such as 8011) to a specific file (such as Event_Settings.htm).

➢ To edit a map file:

1. In RoboHelp, go to the Project Set-up pod. By default, this is accessed from a tab beneath the Project Manager pod.
2. Open up Context-Sensitive Help > Map Files and click on a map file to edit. The ICM help is in the file `Help_Admin.h`.
3. Create map IDs and assign topics to them as described in the RoboHelp help.

The mapping process produces two files:

- `.h` - This file maps friendly names (Topic IDs) to map numbers. A project can have multiple map files.
- `.ali` - This file maps Topic IDs to filenames. A project can only have one alias file.

Ensuring the code calls the correct help system

Just as an FYI, in case something breaks in the future and we need to explain to a new developer how this all works, here's how the help is called from the app:

- The property that controls which help system is called `is baseHelpUri` in the file *MAIN*`/devel/content/src/Urls.properties` (where *MAIN* is whatever branch the code is in).
- This property is in turn used by the `mercedBaseHelpUri` variable in *MAIN*`/devel/content/src/frag/new_start_frag.jsp`.
- The `mercedBaseHelpUri` variable is used by the `mercedOpenHelp` function in *MAIN*`/devel/documentation/src/javascripts/MercedHelpLib.js`, which uses the `mercedBaseHelpUri` variable plus the doc-key value from the cbean of the current page to construct and open the URL of appropriate help topic.
- When packaging the product, this property is overridden in the product layer (*MAIN*`/mps` instead of *MAIN*`/devel`) as follows:
  - Sales PM: *MAIN*`/mps/products/spm/src/Urls.properties`

Using FrameMaker files in help

The FrameMaker files used in both the NICE PM and NICE ICM help are in separate book files. The conversion process can preserve links between FrameMaker files only if the files are in the same book when they are converted.

The books are then *linked* into RoboHelp rather than *imported*. Linking books means that RoboHelp detects when the files change, and that the topics can be updated without reimporting the files.

Setting up topic breaks in FrameMaker

The topics derived from FrameMaker source files use Filename markers to specify the file names and TopicAlias markers to specify the context-sensitive aliases. Because of these markers, we cannot split chapters into topics at the format level. For example, typically when you set up the conversion mapping, you can say that you want every Heading 1 and Heading 2 to be a new topic but not Heading 3s. Instead, every time RoboHelp encounters a Filename marker, it creates a new topic. This means that you use Filename markers in FrameMaker only on headings you want to start new topics, but continue to use the TopicAlias markers for headings you might want to jump to from the Help link in a specific screen.

Most new topics start at Heading1. However, if you do want to start a new topic at a Heading2, it's going to end up looking wrong in RoboHelp (the topic will start with the Heading 2 style instead of the Heading 1 style). This problem is not yet solved.

The solution might be to come up with a pre-processing script that promotes any Heading2s to Heading1 if they contain the Filename marker, but someone needs to look into this to see whether this is possible. The alternative is to add more styles to the template similar to what was in the old template that allows you to specify that something looks like a Heading2 in Frame but then becomes a Heading1 in RoboHelp, but this clutters the template.

Working with linked files in RoboHelp

This section explains how to create RoboHelp topics using linked FrameMaker files.

Linking FrameMaker files

Before linking any files, ensure you are using the standard NICE file conversion settings.

➢ To import conversion settings into a RoboHelp project:

1. In Perforce, sync to `TECHPUB\resources\help`.
2. Select **File > Project Settings**.
3. On the Import tab, click **Browse**.
4. Navigate to `TECHPUB\resources\help`.
5. Select the RoboHelp Import Settings File (`.isf` file) and click **Open**.

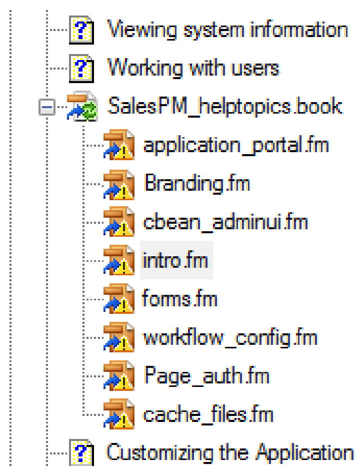
These settings specify the mapping of FrameMaker styles to RoboHelp styles, how single .fm files are split into multiple help topics, and other aspects of the conversion from FrameMaker to RoboHelp.

➢ To link files:

1. Click **File > Link > FrameMaker Document**.
2. Navigate to the FrameMaker book file to import, and click **Open**.

   The book file must be closed in FrameMaker before you can open it in RoboHelp. Make sure you choose **Files of type: FrameMaker Book** to see the available book files.

The imported book appears in the Project Files folder as a distinctive icon with blue and green arrows, with a name of *filename*`.book`. Click the + sign to see the files contained within the book. These act as folders containing individual help topics for each subheading:

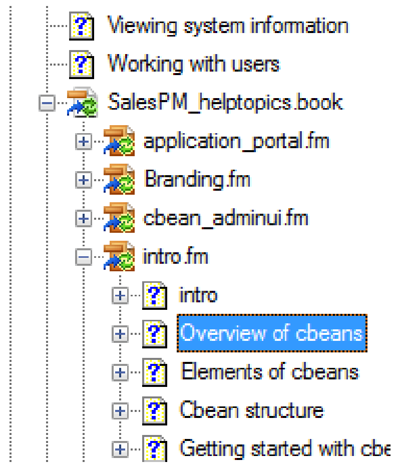


After you link the files, you can create help topics from them. This is a separate step.

➢ To create help topics from linked files:

1. In RoboHelp, right-click the book file.
2. Select **Update > Generate**.

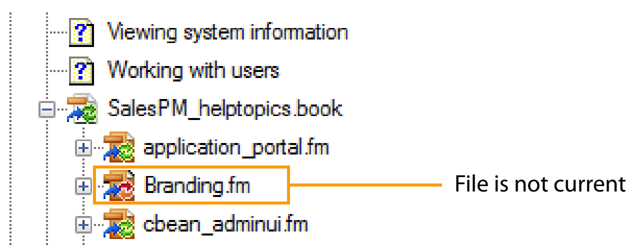
When RoboHelp is finished, each FrameMaker file should contain a set of help topics. These can be added to the table of contents and index, or linked to by other topics, in exactly the same fashion as other RoboHelp topic files.



WARNING: Do not directly update topics created from linked files. Your changes will be overwritten when the files are updated. Make any necessary edits to the source FrameMaker file instead.

Updating linked files

To synchronize the help topics with changes made to the FrameMaker files, the files must be periodically updated. If a file changed since it was last synchronized, its icon indicates that it is no longer current.



➢ To update linked files:

1. Back up your ToC file (`.hhc`) to a different folder.
2. Select the linked file or book that needs to be updated.

   WARNING: If RoboHelp crashes when you update your book, you might have to update each file in the book individually. It might be that too much memory is required to update all the files at once. Also, be sure to shut down every app you don't need before you start this process (see "Close all unneeded applications" on page 133 and "Preventing crashes" on page 139)).

3. Right-click and select Update > Update All. The help topics for the file will automatically be regenerated.

4. If relevant topics have been added to or deleted from the linked files, manually update your ToCs to include or remove them.

NOTE: Rarely, RoboHelp deletes the help topics instead of regenerating them.

➢ If RoboHelp deletes your linked topics:

1. Close RoboHelp without saving your project.
2. When prompted to save your changes to the table of contents, choose not to. If you do accidentally save the table of contents, overwrite the file with your backup before proceeding to the next step.
3. Reopen the project and attempt the update again. Do not worry about all the broken links. If the files update correctly this time, they should reappear in the ToC automatically and the broken links will disappear.

NOTE: If you save the changes to the table of contents and do not have a backup, you will need to revert to the most recent table of contents file in source control.

Updating forms.fm

The shared forms.fm file has a really annoying problem. In the Specifying a data source web service topic, a table that's conditionalized out of Service PM still shows up in the Service PM help. To work around this bug, edit the conditional tag in the HTML in RoboHelp. Therefore, whenever you update forms.fm, you must manually adjust the conditional start and end tags on the HTML tab.

Working with stylesheets in linked files

When you integrate a FrameMaker file with RoboHelp, RoboHelp creates a separate stylesheet for that file. This is very helpful, because as it turns out, you have to make some manual tweaks to the indents in the bullets and indented body paragraphs for the Frame files that you don't want to apply to the topics authored natively in RoboHelp. The bad news is that Frame then wants to overwrite your individual Frame stylesheets on each update. To avoid this, after making your stylesheet tweaks, right-click the Frame file in RoboHelp, choose Properties, and then on the File Update Settings tab, select the stylesheet. This will prevent it from being updated in the future.

Working with videos in linked files

You can embed a video in a FrameMaker file that's single-sourced into RoboHelp. The problem is that RoboHelp only creates an object tag without the embed tag, which Firefox can't parse. To work around this issue, you must edit the topic in RoboHelp and add the embed tag (see the NICE NICE ICM help topic "Demo of creating a page" for an example). To prevent your manual tweak from being overwritten the next time you update the source FrameMaker file, right-click the Frame file in RoboHelp, choose **Properties**, and then on the File Update Settings tab, select that topic so that it's no longer updated.

Conditionalized cross-references

If you notice that a cross-reference that you're trying to tag as Not in Help is still showing up, it's probably because the FrameMaker to RoboHelp conversion setting for that xref type is set to display the RoboHelp paratext instead of source. This means that when converting the file from frame to RoboHelp, it converts the cross-reference to text in the current RoboHelp paragraph style, which sadly deletes the conditional text setting.

Cross-refs to chapters and appendixes should have their conversion settings set to [source] so that they can be conditioned out. If you want to create a cross-ref to a chapter or appendix, make it to the first topic in that chapter/appendix instead.

Generating an index from FrameMaker index entries

When you link a FrameMaker file or bookfile instead of importing it, the only option available to you for generating the index entries is to add them to a new index file. However, you can then add that index file as a placeholder in the existing index, allowing you to blend the index created in RoboHelp and the one created from FrameMaker. For more info, see the post on the Adobe forum at:

http://forums.adobe.com/thread/781364

Updating linked files after branching

After you branch the FrameMaker manuals and RoboHelp project, re-link the linked files as follows:

1. Back up all RoboHelp table of contents (`.hhc`) files. What you are about to do will remove all the linked topics from the ToC, and your life will be easier if you can just restore the original version afterwards.
2. Male sure the FrameMaker book you use for your linked files contains the latest versions.
3. Open the project file in RoboHelp.
4. Remove the existing linked files from the help project by deleting `*Help.book` from the Project Manager. This removes all the topics based on them from all ToCs.
5. Link the new version of the FrameMaker book and generate the topics based on them (as described in "Linking FrameMaker files" on page 127).
6. Copy your backed up ToC files back into the main project.
7. Check that all the topics appear correctly in the ToC.

Sharing content between help projects with linked FrameMaker files

If you want to be able to create cross-references between FrameMaker files, the files must be in the same book. So the correct thing to do is to create a container book that has all the files you want for NICE PM, for example, and then bring that book into RoboHelp as a linked resource. However, if we have one book for NICE ICM and one book for NICE PM, and each of those books point to some of the same files (for example, `forms.fm`), we can't pull both those books into the same RoboHelp project, because you can only have a FrameMaker file appear once in the project. Therefore, each product needs its own RH project, and each will have a corresponding FrameMaker book that includes all the FrameMaker files needed by that help system.

Another problem is that we have several topics authored in RoboHelp that need to be shared by both help systems, but you can't use the same topic in more than one project. One solution might be to create a separate project for these shared files and import it by reference into each of the master projects. However, this presents problems for cross-references, because you can't create links to files outside of your project's directory structure. One approach would be to try to put all projects in the same root folder, but I don't think this will work, and I'm loathe to attempt it, considering RoboHelp already crashes regularly with normal use. The solution, then, seems to be that we should move the shared topics that are currently in RoboHelp into a FrameMaker file and include that file in each of the container books. We just need to be sure to conditionalize the text in the shared FrameMaker files for the two products, but we're already in the habit of doing that.

Down the road, if we need to create a merged help system of both NICE ICM & NICE PM, I'm guessing we can easily create a simple uberproject that has a table of contents that points to the two help projects. Each of those projects would need to have a table of contents designed specifically for this uberproject. So for example, since there's info on forms in both projects, we could make sure the uberTOC in the NICE ICM project does not include the forms topic and make sure the uberTOC in the NICE PM help project does, or vice versa. I have no idea what kind of special breakage will occur, especially with regard to conditional text settings; we would need to experiment with this further.

Generating output

This section describes how to generate the help and *User Guide* from the NICE ICM and NICE PM RoboHelp projects.

Generating the help

Sync and check in first

Before you begin, always sync to the latest files by opening Perforce, selecting the TECHPUBS directory on the Depot tab, and then right-clicking and choosing Get Latest Revision. You should also check in any files you currently have checked out so that Perforce has the latest clean set of files in case anything gets corrupted during generation.

Close all unneeded applications

RoboHelp takes up a huge amount of memory when you update linked files or generate the help, and it crashes easily if anything else is running, which usually results in corrupted files.

If you find you tend to have problems when generating the help, shut down every application you possibly can before you start, including Perforce, Outlook, Yammer, Skype, and VPN.

Edit linked files as needed

Make sure the linked FrameMaker files included in the help project are updated and look correct when you apply the conditional text expression for the help you're generating.

Update linked files

For more information, see "Updating linked files" on page 129.

Spot check topics from linked files

After updating, spot check the RoboHelp topics based on imported FrameMaker files to make sure they were imported correctly. Sometimes RoboHelp applies some unexpected formatting. If so, try closing the topic and opening it again. If this does not fix the problem you must either remiport that file or make a manual fix.

Generate the first draft of the help output

After you generate the first draft, examine the entire output for formatting issues, linked doc issues, variable issues, and plain text that should be hyperlinked under see also / related topics. Also search for terms that shouldn't appear, such as previous product names. You will need to manually tweak the files described in the following section.

Manually fix formatting and numbering

If there are multiple procedures in a single linked topic, the numbering does not restart at 1 for each procedure. To work around this, you must manually add `start="1"` to the `<?rh-list_start class="Numbered" level="1" ?>` element following the procedureheading line. For example:

```
<p class="procedureheading"><a name="IX_editing_DICE_monitors"></a>To
edit
 an existing DICE monitor:</p>
<?rh-list_start class="Numbered" level="1" start="1" ?>
```

Following are the NPM topics that currently have multiple procedures:

- Taking actions on tasks
- Using report pagination controls
- Coaching session workflow (three procedures)
- Coaching based on observations
- Searching for best practices
- Managing best practice categories (three procedures)
- Managing best practice content items (several procedures)
- Exporting configurations
- Import command
- Creating status metrics
- Adding filters
- Sorting the report's data
- Creating prompts on the dashboard authoring screen
- Using the content tree
- Publishing content items
- Working with profiles
- Working with profile criteria
- Configuring the TimeDimensionPopulator
- Adding an entry to the CustomTimeGrainSeed table
- Adding new axes
- Creating namespaces
- Promoting employees
- Demoting employees
- Working with cube aggregates
- Moving base metrics
- Changing File Sniffer directories or DICE jobs

Additionally, because RoboHelp tends to add a blank numbered step after tables and cause other formatting issues, you must modify the following topics manually:

- Using temporal modes in reports (stray bullet after table)
- Swapping axes (line break in step 3)
- Creating metric sources (line break)
- Editing reports while viewing content as another user (delete link to Configuring access at end of topic)
- Setting permissions (indent the text under the indented bullet)
- Prompt and content interactions (stray bullet)
- Specifying a data source web service (delete ref to configuring web services, delete table)
- Adding components to a page (stray bullets)
- Applying permissions to content
- Adding new groups
- Adding new persons to the Hierarchy Manager

Test output on an actual server

If possible, test the output help on an actual server (such as DocDemo) prior to the help freeze deadline. Some problems (such as backslashes in the stylesheet path) won't show up on your own system, but will cause the help to break when deployed.

The locations to copy the output help for testing are as follows:

Component	Help location
NPM Core Server	Within the `webapps` directory of NICE PM at the following location: `npm/webapps/merced/documentation/com/merced/documentation` Copy the complete `SalesPMHelp` directory (for NICE ICM) or `ServicePMHelp` directory (for NICE PM) to this location.
ICM Admin	Within the deployed ICM user interface (`micm`) web application at the following location `apache-tomcat-7.0.27/webapps/micm/onlinehelp/adm/en` Copy the contents of the `SalesPMHelp` directory to this location, not the parent `SalesPMHelp` directory.
Liferay	Within the deployed nice-documentation web application at the web server running Liferay at the following location: `liferay/apache-tomcat-7.0.27/webapps/nice-documentation/com/merced/documentation` Copy the complete `SalesPMHelp` directory for NICE ICM or `ServicePMHelp` directory for NICE PM to this location.

Check in final output

The final version of the help output should be checked in two weeks before GA.

You will generate the help many times. Well before code freeze, after verifying your final output is working on the server with no issues, check the files into the source directory and to the target directory in Perforce.

The directories to use for the output files are:

- NICE ICM. For the MAIN branch, use this directory:

  `//unified/MAIN/mps/devel/documentation/src/com/merced/documentation/SalesPMHelp`

  For a specific release, use this directory:

  `//unified/releases/NICEPM/`*version*`/mps/devel/documentation/src/com/merced/documentation/SalesPMHelp`

- NICE PM: ***

Make sure you right-click each directory and choose Add Files to add any new files you might not have checked in previously. Test the help in the daily build to make sure nothing broke, and then let QA know that the final help is checked in.

Generating accessible help

The accessible single source layout is the same as the standard WebHelp. The only difference is that in the Navigation settings, the Section 508 Compliance Output box is checked.

If your help project includes topics created from linked FrameMaker files, then after generating the accessible help you must add `scope` tags to the tables created in FrameMaker (see "Accessibility" on page 122 for more information on heading scope).

NOTE: You can also add the scope tags the RoboHelp source, but if you update the linked files afterwards your changes will be lost. Whether you choose to edit the source files or the output files, do this step at the very last minute.

The easiest way to make this change is a multi-file find and replace using Helpware Find and Replace (FAR). Find text that meets the following criteria:

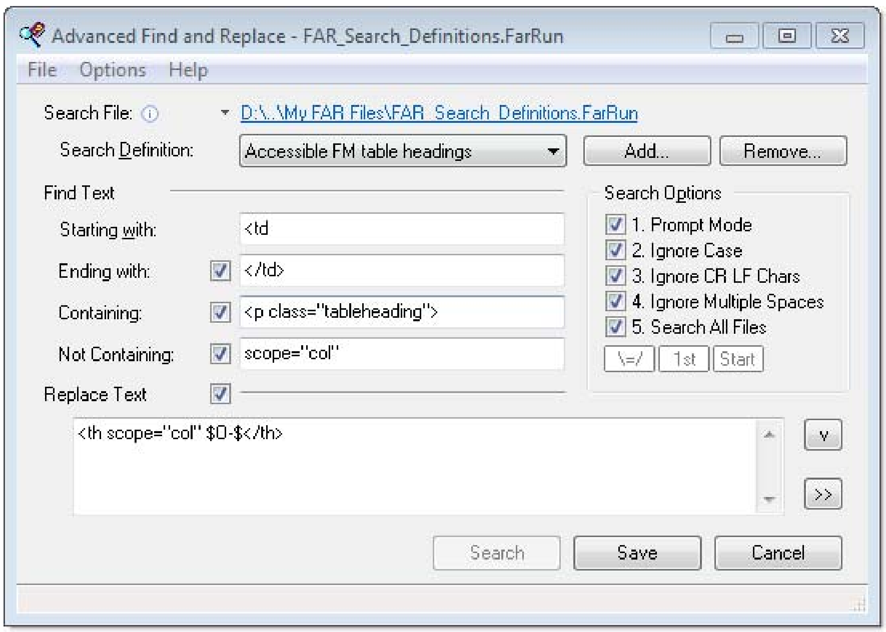
- Begins with `<td` and ends with `</td>` (the FrameMaker to RoboHelp conversion uses the `<td>` tag even for header cells).
- Contains the string `<p class="tableheading">` (the FrameMaker TH_TableHeading style is converted to this RoboHelp style).
- Does not contain the text `scope="col"` (this ensures that RoboHelp tables that are already correctly formatted are not changed).

Your search should then replace the start and end text with <th scope="col" and </th> and leave the text in the middle unchanged. In FAR, this is achieved using a Replace Text of:

```
<th scope="col" $0-$</th>
```

NOTE: This find and replace method assumes that all table headings in FrameMaker documents are column headings. If any of your table headings are row headings, you must edit those tables manually.

The Accessible FM table headings script in FAR updates the scope:



A copy of the search definitions file containing this script is checked into Perforce here: TECHPUBS\resources\help\FAR_Search_Definitions.FarRun

About the WebHelp format

The WebHelp format is based on HTML. Each of the topics is held as a separate HTML file.

Table of contents format

In the unlikely event that you (or CSG) need to manually add or remove files from the output help, you may need to know about the structure of the table of contents.

The table of contents is stored in the whgdata folder. It consists of a set of files, each of which contains a set of links that represent the tree down to a specific subfolder. Some of these links are to topics, others are to other contents files. From a near top-level file (for example whstt5.htm for NICE ICM), you can use these links to find the contents file for a specific section.

Generating the NICE ICM user guide

Use the Printed Documentation single source layout to generate the NICE ICM *User Guide*.

NOTE: In the future, instead of generating the NICE ICM *User Guide* from RoboHelp, we might generate it from FrameMaker if we pull all the source content into FrameMaker.

Enable Word macros for PDF output

If you choose to generate PDF output directly rather than going through Word (for example, to check how something will work in the printed help), you might get a Word document template window with an error. This is because RoboHelp produces PDFs via Word's normal.dot template, and the process requires running a set of macros. From the template window set normal.dot to always run macros from this publisher (or something along those lines) and try again. It should work.

Update the Word index before generating PDF

The index generated for Printed Documentation might be incorrect by a couple of pages. I don't know why. Updating it seems to solve the problem.

Generate the guide

➢ To generate the user guide:

1. Get the latest Word template (.DOT) from the resources folder in Perforce.
2. Generate the guide and then do the following:
   a. Check the Table of Contents in the Single-Source Layout. Remove any legacy topics remaining in Chapter layout, and add any missing topics.
   b. Check the mapping of styles.
3. Search resulting Word document for terms that shouldn't appear. (They might appear thanks to the non-updating of the Printed Documentation ToC even if all was fine in the help.)
4. Check the document for topics that shouldn't exist.
5. Generate the final version.
6. Enable and run the macros for table formatting and xrefs.
7. Fiddle with any styles necessary to make the indents correct. (Sometimes the bullets will not indent correctly. Currently the only fix known for this is to set the margins to 5cm, which is where the bullets should be, and move all the other styles left as necessary.)
8. Select and update all the text that claims to be in the Normal style but isn't.

9. Make manual adjustments:
   a. Move wide tables left.
   b. Reverse any inversed bolding in bulleted lists (the reason for this is unknown).
   c. Fix bullets in FM topics.
   d. Miscellaneous formatting fixes.
   e. Change headers/footers to use doc title.
   f. Check cover and add correct date.
10. Set Section Start to New Page (rather than the default Odd Page) and apply to whole document. (Page Layout > Page Setup > Layout tab).
11. Update all fields (Select All, F9).
12. On the **Acrobat** tab, click **Create PDF.** Any other method of creating a PDF from Word risks being huge or containing no bookmarks.
13. Check for miscellaneous blank pages.
14. Check the ToC.
15. Regenerate if necessary.
16. Remove > characters from PDF ToC.
17. Apply the standard PDF settings described elsewhere in this document.
18. Add the *User Guide* to the portfolio with the other PDF files.

Troubleshooting

This section describes some RoboHelp issues and how to resolve them.

Preventing crashes

Both FrameMaker and RoboHelp can take up a lot of a computer's resources and can cause your computer to crash or hang. Before updating FrameMaker files in RoboHelp, close Perforce, Outlook, Skype, and anything else that might take up needed resources and cause RoboHelp to crash.

The project cache file (`.cpd`) is also notorious for causing crashes. To prevent them, configure RoboHelp to clear the cache before opening a project as follows:

1. In RoboHelp, choose **Tools > Options**.

2. On the General tab, select **Clear project cache...** and then click **OK**.

NOTE: If your FrameMaker files are not showing up as expected in RoboHelp when you open it (for example, the book file is missing, or the topics under a Frame file are missing), or if you get errors that the APJ file has errors, the solution is to turn off the option to clear the project cache.

Fixing corrupted projects

Signs of a corrupted project include the project opening very slowly, your pointer flickering between arrow and hourglass, topics broken or missing from the topic list, or RoboHelp crashing when you attempt to save. This is caused by the `.cpd` file.

If you think your project is corrupt, follow the following steps:

1. Back up your project (for example, by copying it to a new folder).
2. Delete the `.cpd` file.
3. Open the `.xpj` file.

RoboHelp will recreate the `.cpd` file based on information from the other files in the project.

Context help does not work

Check that the help mapping is correctly set up in RoboHelp. For information on help mapping, see "Implementing context-sensitive help in NICE applications" on page 124.

If there is no mapping information present, but topics were mapped correctly before, this normally indicates that information has been lost from the `.h` and `.ali` files. Replace these files with the last correct versions from Perforce.

Rarely, although all help mappings are correctly set up, none of them function in the output help. This is because RoboHelp has failed to correctly generate the following files:

- `cshdat_robohelp.htm`
- `cshdat_webhelp.html`
- `whcshdata.html`

The files are present in the RoboHelp output, but contain no file mapping information.

In this case, the solution is to copy the mapping information from the latest previous version of the help output in which these files were correctly generated.

There is no known way of forcing RoboHelp to generate these files correctly, but the problem may spontaneously resolve in later versions of the project.

"Show Table of Contents" link missing from context help

When a topic is called from the application UI, there is no Show Table of Contents link in the topic to enable users to access the full help system. To fix this:

1. Check the output version of the topic that should be appearing. Compare it to the output version of a topic that does work. You might find either that there are scripts missing from the problem topic, or that it contains scripts that are not in the working topic. These are probably the problem.
2. Open the source for this topic in RoboHelp for edit, and fix the problem as described in the "Missing Scripts" and "Extraneous Scripts" sections.

Undefined variable error message

If you change variables outside RoboHelp - for example, by doing a search-and-replace in FAR - RoboHelp will not correctly update the variables in each topic. Everything may appear properly in the WYSIWYG, but you will be deluged with errors in the output.

If you're finding these errors in your output and everything looks correct in your code, try manually opening each file containing these errors, making a minor change, and re-saving. This should force RoboHelp to update the variables properly.

Missing Scripts

Rarely, some topics in RoboHelp may not include scripts that appear in most other topics.

This is caused by RoboHelp not updating files properly, similar to "Undefined variable error message." Make a minor change to the topic and save it. Generate the help again, and RoboHelp should have mysteriously added the necessary scripts.

Extraneous Scripts

Rarely, some topics in RoboHelp contain unnecessary scripts that do not appear in most other topics.

This is probably a problem with links within the topic. Check all the links and remove any extraneous information such as "id" tags. When all the extraneous tags have been removed, generate the help again, and you should get the correct scripts. Don't ask me why this happens.

Error processing RoboHelpFrameDocs.apj

If you get this error, the RoboHelpFrameDocs.apj file is corrupted. Search through the file for instances of

```
<genfile/>
```

and delete all instances. Make sure you don't delete all the genfile tags, just those that are a self-closing version of the tag. Also, this corruption happens when you've set the option to clear the cache each time RoboHelp opens, so in this case you'll want to change that option back to the default of not clearing the cache (see "Preventing crashes" on page 139).

9 Distributing Documents

This chapter contains the following sections:

NOTE: Before publishing any documentation, be sure to sync to the latest PDF files in Perforce.

Sending PDF drafts for review

➢ To send out a PDF draft:

1. In FrameMaker, choose **File > Save as Review PDF > Send through email**.

   This option includes a Send button in the PDF for returning comments.

2. In the email, replace the default message with something like this:

   ```
   Attached is the latest draft of the <topic>. Please mark up the PDF
   and send it back. I'll incorporate your comments and then send out
   the entire chapter for final review.
   ```

3. Click Send.

Publishing documentation on the twiki

This section describes how to publish new documentation on the documentation twiki for internal use. The Merced Systems Twiki (http://intranet.mercedsystems.com/twiki/bin/view/Merced) contains a Technical Publications section the contains links to the current documentation, all released documentation, and resources (such as this style guide).

Question/Issue: TBD: this section plus brief section on publishing style guide

Publishing documentation on the merced drive

When the documentation is ready for distribution (typically the day before the GA date), you publish the doc set on the `merced` network drive, also called the I: drive.

➢ To publish the documentation on the network drive:

1. In Perforce, sync to the latest PDF files in the release folder.

   For example, `//eng/TECHPUBS/NPM/<release>/pdfs/<release>/`.
2. Select one of the files, right-click, and choose **Show in Explorer**.
3. Press CTRL-A to select all the files in the folder.
4. In the Computer section of the left navigation panel, click the I: drive.
5. Navigate to `I:\Public\Tech_Pubs\Published`.
6. Create a new folder. For example: `NPM4.2.0`.
7. Paste the PDF files into the folder.

Publishing a portfolio on the customer portal

After final approval and production checks, publish the documentation PDF portfolio and individual Captivate training videos to the customer portal on salesforce.com.

Normally, publish the single PDF portfolio file and not individual documents.

For videos, create a ZIP file (flat with no directory structure) containing the published output from Captivate: *filename*`.htm`, *filename*`.swf`, and `standard.js`.

➢ To publish a file to the customer portal:

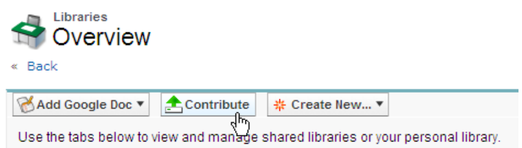
1. Go to https://login.salesforce.com/ in any browser.
2. Log in using following:

   User Name: `jackie.wheeler@mercedsystems.com`

   Password: `TechPubs`

3. Click **+** on the tab line.

4. Click **Libraries**, then **Contribute**.



5. Click **Choose File,** select the PDF portfolio to upload, and click **Open**.

   When you click **Open**, SalesForce uploads it to a temporary file area. The upload can take several minutes.

6. After the file has been uploaded, enter the information describing the file

   a. For **Title**, type a suitable title. This title is the text that appears to users of the customer portal when they browse the library. Use a title consistent with the existing entries. A PDF portfolio title should contain the product name and version and end in "Documentation." A video title should end in "video." For example:

      Sales PM 4.1.0 Documentation

      Creating Pages video

   b. For **Description**, type a brief description of the file.

   c. Click **Publish to a shared library** and select **Portal documentation and Training**. Wait a few seconds and additional options described below appear.

   d. For **Tags**, enter the tags for the document. You can either type new tags or click the entries under **My Recent Tags** to enter the tags. Separate tags by a comma. Include at least the following tags:

Product	Tags
NICE ICM 4.2.1	`NCIE ICM 4.2.1, NICE ICM`
NICE PM 4.2.1	`NICE PM 4.2.1, NICE PM`
MPS	`MPS 3.8`
ICM	`ICM 3.2`
Videos	`videos`

e. For **Document Type**, select one of the following values:

Document Type	Description
Product Manual	`PDF document`
Training Video	`Captivate video`

f. For **Last Review Date**, click the link next to the field to populate the field with today's date.

g. Ignore the other fields.

h. Click **Publish** to publish the document, or **Delete** if you decide you don't really want to publish. It is Delete rather than Cancel because the file has already been uploaded to SalesForce even though not published yet.

i. Click **Done Publishing** to close the window.

➢ To verify that the portfolio is available:

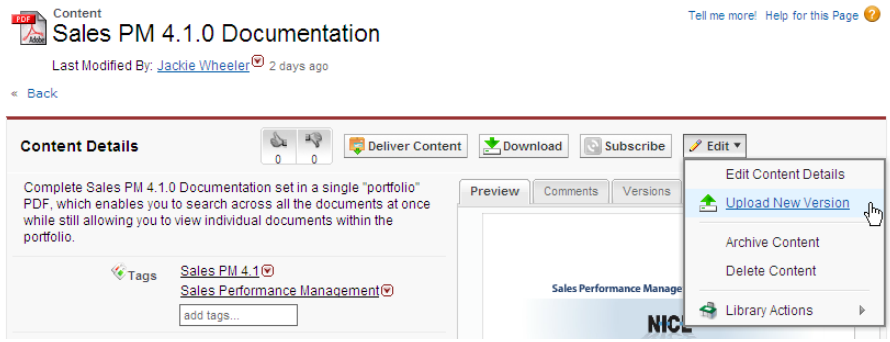
1. If the list of libraries is not already displayed, click **All Libraries**.
2. Click **Next** at the bottom of the **My Libraries** list.
3. Click **Browse** next to **Portal Documentation and Training**.

   It can take a few seconds for new files to appear at the top of the library list.

➢ To update an existing library entry:

1. Log in as normal.
2. Open the details for the entry you want to update. For example:
   a. Click **+** on the tab line.
   b. Click **Libraries**.
   c. Click **Next** at bottom of **My Libraries**.
   d. Click **Browse** next to portal Documentation and Training.
   e. Click the **Title** of the file.
3. To change the file content:

a. Choose **Edit > Upload New Version**.



b. Click **Browse** to upload the new file.

c. Complete the Information dialog box with any changes.

d. Click **Save**.

4. To change the descriptive information:

   a. Choose **Edit > Edit Content Details**.

   b. Make changes as required.

   c. Click **Save**.

Publishing a portfolio on the EFT server

This section describes how to log in to the EFT server and upload the portfolio PDF. This procedure requires that you have an EFT server account. These are not the same as your Windows credentials or the read-only credentials shown in the introduction.

If you don't have an EFT server account or forgot your password, create an issue in MIRA. Select `Global Customer Hosting (no labs)` as the Project and `Support Ticket` as the Issue Type.

The following read-only credentials are used by customers to download the portfolio:

Username: `docs`
Password: `MY_m3rc3d`

➤ To publish a portfolio on the EFT server:

1. In your browser, go to **Merced Web > WebHome** and click **File Upload Service**.
2. Click the URL (https://eft.mercedsystems.com).
3. Enter your EFT server username and password.
4. Click the **Docs** folder.
5. Click **New Folder** and specify the folder name (for example, `NICEPM 4.2.0`).

6. Click **OK**.
7. Select the folder.
8. Click **Upload**, and then click **Choose File**.
9. Navigate to the `Published` folder on the I drive (`\\merced\Public\Tech_Pubs\Published`), or to the folder in Perforce, and open the folder you want.
10. Select the portfolio PDF and click **Open**.
11. Click **Upload**.

NOTE: Do not upload PDFs for the individual documents.

Creating the announcement on MUSE

After you publish the portfolio on Salesforce.com and on the EFT server, post an announcement on MUSE.

➢ To create the announcement:

1. Go to **Merced Web > WebHome** and click **MUSE**.
2. Log in using your MUSE username and password.
3. Scroll down the page and click **MPS Documentation**.
4. Click the latest announcement about new documentation available.
5. Select the text and copy it.
6. Click the back button in your browser to get back to the MPS Documentation page.
7. Click **New Topic**.
8. Paste the content in the body field and update versions and URLs as needed.
9. Enter the subject: `NICE PM <release> documentation now available`.
10. Click **Save**.

Documentation review process

This section describes the basic steps in the documentation review process at NICE.

1. Save the FrameMaker or book files as a PDF.
2. Open the PDF file in Adobe Acrobat and choose **Comments > Enable for Commenting and Analysis in Adobe Reader**. This saves the file in a form that enables reviewers to add comments using Adobe Reader.

3. Post the PDF to the appropriate release folder in the Drafts public folder.

   The full path for this folder is `http://intranet.mercedsystems.com:8080/public/TechPubs/Drafts`.

4. Add a link to the Twiki doc review page for the project.

   For example, for the 3.0 release, the doc review page is `http://intranet.mercedsystems.com/twiki/bin/view/Merced/EinsteinDocsForReview`

5. File MIRAs for each Development and Product Management reviewer using the following settings:
   - Feature Category = Documentation_Training
   - Due Date = date the review comments are due back
   - Description = include a link to the document saved in the public folder

6. Send an email with the pertinent information, such as the due date, to all required reviewers and other interested parties.

7. To keep the other writers in the loop, copy the techpubs Outlook alias on all review emails.

Documentation release process

This section describes the basic steps in the documentation release process.

1. Complete the `ProductionEditChecklist.doc` stored in the Source folder.

   The full path for this folder is `//eng/MAIN/doc/resources/book_templates`.

2. Save the book file as a PDF:

   a. Choose **File** < **Save Book As**.

   b. Enter the name of the PDF in the **File name** text box.

   c. Choose `.pdf` from the **Save as type** drop-down list.

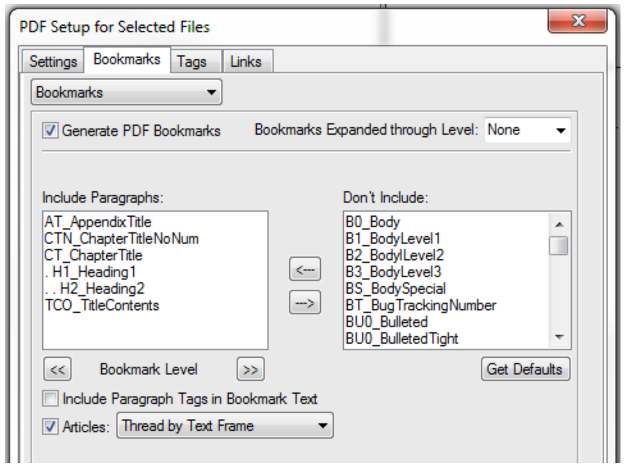
   d. Click **Save**.

3. In the PDF Setup dialog box, click the **Bookmarks** tab and set the bookmarks that are generated in the PDF as follows, and then click **Set**.

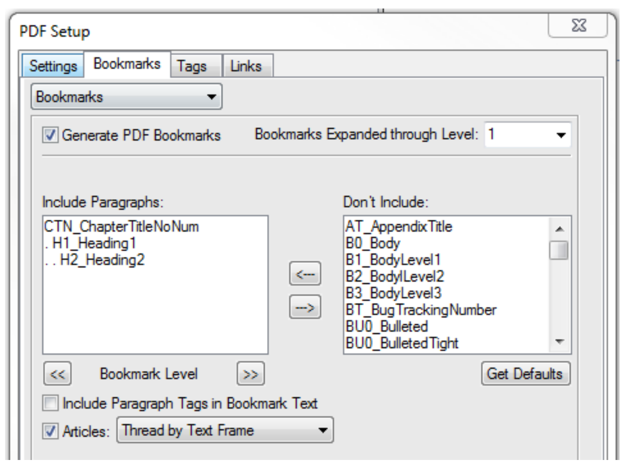
   The paragraph tags included as bookmarks depend on whether you are creating a PDF from an entire book or from only one document.

   - For a book file, move the following paragraph tags into the **Include Paragraphs** box on the left and set them to the specified bookmark levels using the left and right arrow buttons:
     - (Top level) TCO_TitleContents (the paragraph tag used on the cover), AT_AppendixTitle, CT_ChapterTitle, and CTN_ChapterTitleNoNum
     - (Second level—preceded by one period) H1_Heading1

- (Third level—preceded by two period) H2_Heading2



- For a single file, such as a tech note, move the following paragraph tags into the **Include Paragraphs** box on the left and set them to the specified bookmark levels using the left and right arrow buttons:
  - (Top level) CTN_ChapterTitleNoNum
  - (Second level—preceded by one period) H1_Heading1
  - (Third level—preceded by two period) H2_Heading2



4. Set the following Document Properties for the PDF file (**File** > **Document Properties** in Acrobat):
   - **Description** properties:
     - **Title**: *long name of product - booktitle* (do not include the version number)
     - **Author**: NICE Technical Publications
   - **Initial View** properties:
     - **Navigation tab**: Bookmarks Panel and Page
     - **Page layout**: Single Page Continuous
     - **Magnification**: Fit Height

- **Open to page**: 1
- **Show:** Document Title

5. Post the PDF file in the public Published folder (or Drafts if it is a draft for review).
6. Create a Twiki link for the PDF.
7. For an n.n release (NOT a patch)~~, add the PDF to the MPSn.n_TechDocs zip file, which lives in Published and is linked to Complete_Release_Documentation.~~
8. For an *official* release, ~~edit the Product Releases Twiki and add the link to the PDF in Published to the appropriate release. You can do this by copying and pasting the link you want from previous releases and simply changing the filename.~~
9. Send an email to ~~mps30-users~~ that the docs are available on the Product Releases page along with the product jar, and on our doc Twiki as well.

Release guides

A release guide should accompany every release and hotfix. This includes new release features, known and fixed issues, and upgrade information.

- For a major release (such as 4.0 or 5.0) create a Release Guide that describes new features in this major release. There should be no 'fixed issues' section.
- For a minor release (such as 4.1 or 4.2) create a Release Guide that describes the new features in this minor release, the issues fixed in this release, and the known issues cumulative with the last major release (since the clients will find them anyway!).
- For a maintenance release (such as 4.1.1), create a Release Guide that describes new features cumulative with the last minor release (in this example, new features from both 4.1 and 4.1.1), issues fixed cumulative with the last minor release, and known issues cumulative with the last major release.
- For a hotfix (such as 4.2 hotfix), replace the existing Release Guide for this release with one that includes the issues fixed in the hotfix. A hotfix is issued not as a patch but as a new full install that replaces previous hotfixes and includes cumulative fixes. The new Release Guide should be placed in Public\Tech_Pubs\Published*release* and added to the portfolio PDF.

Known and fixed issues

In theory, lists of known and fixed issues for incorporating in the release guide will be provided to you by PM. In actuality, you will need to create these lists for yourself and then have them checked by PM and QA.

Known issues

To search for known issues in the new release, search for MIRAs with a **Found In Version** of the new release and that are either not fixed or have a **Fix Version/s** of a later release. For example, to find new known issues for 4.1.1 you would search for issues found in 4.1.1 that have a resolution of **Unresolved**, **Won't Fix** or **Deferred**, or that have a resolution of **Fixed** but with a **Fix Version**/s of 4.1.2 or 4.3.

It is often simplest to use multiple MIRA filters rather than to cover all the possibilities in one.

As a general rule, you should include all the P2s and P1s (although the product is normally not released with unfixed P1s), and any P3s that have either been found by an LA client or that will be quickly found by most clients (these are usually minor UI issues).

For each bug you enter in the Release Guide, copy the wording of the Release Guide entry into the Release Notes field on the original MIRA ticket. If you need to reopen the bug to do this, reopen it and then resolve and close it again afterwards.

Fixed issues

Fixed issues are listed by the release they were fixed in. To add fixed issues for a release, simply check in MIRA all the known issues listed in the last Release Guide, and move any that were fixed in the new release to the Fixed Issues section.

Hotfixes

The issues fixed in each hotfix are listed on the twiki at http://intranet.mercedsystems.com/twiki/bin/view/Merced/SPMReleases#4_1_1_Hotfix_Builds

Updating documentation

Question/Issue: TBD

10 Working with Support Tools

This chapter contains information on other tools you use frequently. It contains the following sections:

Working in Perforce

All working source documents are in Perforce for safekeeping and versioning/tracking purposes. The following guidelines apply to Perforce:

- Only source documents (FrameMaker files), final graphics, videos, and important supporting documents are stored in Perforce. PDFs created from final drafts are kept in the appropriate version folder under: `Public/Tech_Pubs/Published/`*version number*.
- Modular guides and tech notes (for example: Internationalization Module, Performance Coaching Module and GUID support tech note) go in the branch for which they were developed. They do not carry forward to new branches unless and until the code is carried forward and/or new features are added.

  NOTE: One notable exception is the 3.0_chart_customization tech note in the 3.5 branch, which was erroneously placed there and later could not be moved back because the EINSTEIN branch was locked.

Question/Issue: The previous bullet is news to me. Since we do so much product renaming and template updating, I have been branching the one tech note so that it doesn't get dropped from the set. Sounds like this is the wrong thing to do.

- On rare occasions, there might be two versions of the same document in a single branch. For example, under MAIN, there are two versions of the app_config guide: 3.0 and 3.2. This happens if Engineering does not branch the code.
- Make change list descriptions brief but meaningful, and update the changelist as you work in the file so that you don't forget what you did. Explain what the changes are and why you made them. List all changes so anybody looking to undo one can locate it.
- Folder naming conventions are: all lower case with no spaces. For example: doc_quick_start or online_help.
- Images are stored in a separate folder under the doc name.

Reconciling new files

While working on a FrameMaker book or Help system, you often need to add new graphics and other files. You must explicitly mark these new files for adding otherwise they will not be stored in Perforce.

You can get Perforce to check if there are any new files on your PC that need to be marked for adding by running the Reconcile Offline Work command.

➢ To reconcile new files:

1. In Perforce, right click over the book or help folder in either the Workspace or Depot tab.
2. Click **Reconcile Offline Work**. After a few seconds a dialog appears listing the files in the book or help folder that are present on your local file system but that are not stored in Perforce.
3. For a FrameMaker book, clear the check box next to any of the `.auto.fm`, `.backup.fm` and `.lck.fm` files. Do not store these files in Perforce.
4. Select the check box next to any files such as graphics that you want to add to Perforce.
5. Click **Reconcile**. Perforce then automatically marks the selected files for adding.
6. Right click the book or help folder and click **Submit** to check in the files to Perforce.

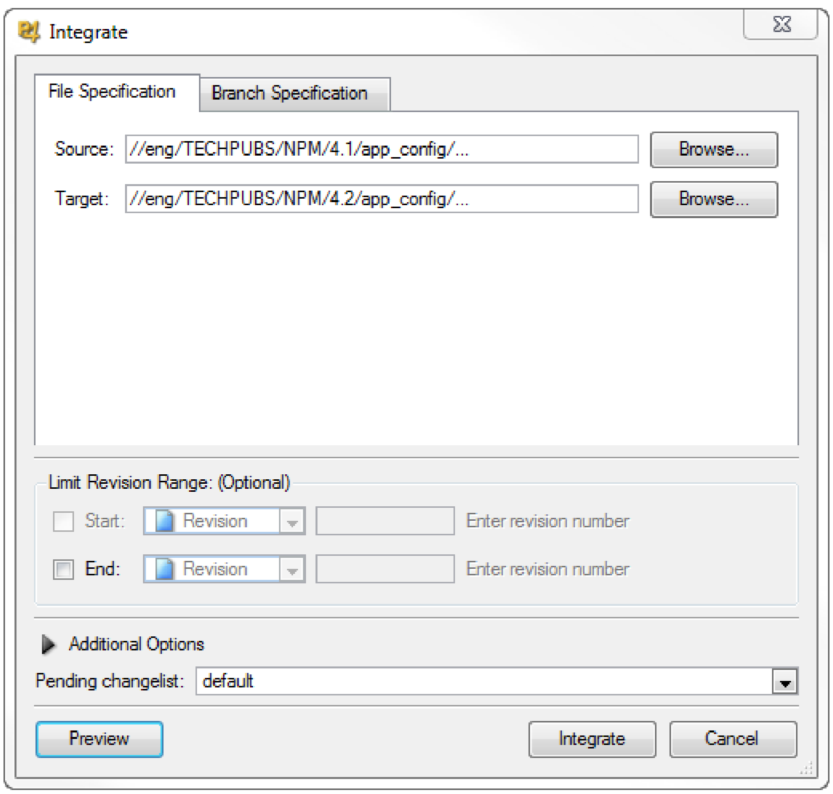
Branching files

If you want to copy a file or directory from one place to another in Perforce, such as from one release to the next, use the Integrate command in Perforce instead of adding the file to the new branch. Branching files preserves their revision history, which is very useful.

NOTE: If you branch a directory and determine that it isn't needed, use **Revert Files** to back out the change instead of **Mark for Delete**. By reverting, you actually delete the files from the Perforce database instead of adding another revision. This helps control the size of the database.

➢ To branch a file or directory:

1. In the depot, sync the file or directory you want to branch.
2. Right click and choose **Integrate**.
3. On the File Specification tab, modify the path in the **Target** text box. Following is an example:



4. Click **Integrate**.
5. When the changelist is populated, click **Submit**, enter a note, and click **Submit** again.

Working in Rally

TBD

Working in MIRA

This section describes how to use MIRA for the following tasks:

- Checking the status of issues that need to be included in the Release Guide
- Routing drafts of issues for review
- Marking an issue as not needing to be included in the Release Guide

Checking the status of issues

TBD

Routing drafts of issues for review

For all known and fixed issues that are flagged for inclusion in the Release Guide, add the text to the Release Notes field right in the MIRA issue. Not only does this facilitate review of the text, but eventually we will generate the release notes directly from MIRA.

So in each release, update this field with the text for that issue and then add the same text in the FrameMaker file. This is an easy way to get the text reviewed; just point the engineer or QA to the MIRA.

Be sure to make yourself a watcher so that you receive a notification if someone provides feedback about your text.

Marking an issue as not needing inclusion in the Release Guide

To mark an issue as not needing to be included in the Release Guide, add the following note:

`xADDx`

This way we can distinguish between issues that need to be considered for inclusion and those that we've already considered and decided not to document.

A Using the Templates

This appendix describes each of the formats in the template set and also provides examples in context.

NOTE: For information on adding documentation templates to FrameMaker, see "Creating a shortcut to the NICE templates" on page 7.

Question/Issue: Appendix needs examples, screenshots

Paragraph formats

This is a h3w

The following table lists the paragraph formats in the NICE FrameMaker templates. Unless otherwise noted, the first column shows text formatted with the specified paragraph format.

Use this tag…	To format …
AT_AppendixTitle (example not shown)	An autonumbered appendix title.
B0_Body B1_BodyLevel1 B2_BodyLevel2 B3_BodyLevel3	In column body text. B0 is flush left and the remaining formats are indented .25, .5, and .75 inches.
BT_BugTrackingNumber	A MIRA number in release guides. For example: ICM-0001, and EIN-0002.
• BU0_Bulleted • BU0_BulletedTight	First level bulleted list items. Use BulletedTight formats only if the list items do not wrap to a new line. If any lines do wrap, use the Bulleted format for the entire list.

Use this tag…	To format …
• BU1_BulletedLevel1 • BU1_BulletedLevel1_Tight	Second level bulleted list items. Use BulletedTight formats only if the entire list is comprised of paragraphs that do not wrap. If any lines do wrap, use the Bulleted format for the entire list.
• BU2_BulletedLevel2 • BU2_BulletedLevel2_Tight	Third level bulleted list items. Use BulletedTight formats only if the entire list is comprised of paragraphs that do not wrap. If any lines do wrap, use the Bulleted format for the entire list.
BW_BodyWide	Body text that runs across all columns and side heads.

`C0_Code`
`C1_CodeLevel1`
`C2_CodeLevel2`
`C3_CodeLevel3`
`C4_CodeLevel4`
	Code samples, such as a command line command or program, a Java code sample, or an excerpt from a non-Java file (such as a configuration file). C0 is flush left and the remaining formats are indented .25, .5, .75, and 1.0 inches. Tabs are .25 inches apart.

CTN_ChapterTitleNoNum (example not shown)	A chapter title with no number. Used for single-chapter documents.
CT_ChapterTitle (example not shown)	An autonumbered chapter title.
`CW_CodeWide`	**Question/Issue: Check this one. Might mistakenly be In Column.**
FIC_FigureCallout	A left-aligned graphic callout paragraph in a text frame inside an anchored frame.
FIC_FigureCalloutCenter	A centered graphic callout paragraph in a text frame inside an anchored frame.
FIT_FigureTitle	A centered figure title. **Question/Issue: Do the titles go above or below the graphic? For accessibility, should be above. FIT calls B0, though.**
FR0_Frame (sample not shown)	A paragraph containing an anchored frame. FR0 sets the space above and below and keeps the paragraph together with the preceding one.
FRS_FrameSideHead (sample not shown)	A paragraph containing an anchored frame aligned with side heads. FRS sets the space above and below and keeps the paragraph together with the next one.
FRW_FrameWide (sample not shown)	A paragraph containing a right-aligned anchored frame, typically for graphics that are too wide for FR0. FRW sets the space above and below and keeps the paragraph together with the previous one.
H1_Heading1	A level-one heading.

Use this tag…	To format …
H2_Heading2	A level-two heading.
H3W_Heading3Wide	A level-three heading that spans columns and side heads. H3W is flush-left with H1. Can optionally be included in the TOC. **Question/Issue: TOC needs to include this tag, not H3R_Heading3RefTerm**
H3_Heading3	A level-three heading. Can optionally be included in the TOC.
H4R_HEADING4REFATTRIBUTE	A level-four heading used in reference sections, such as in command syntax sections. Not included in the TOC.
H4_Heading4	A level-four heading. Not included in the TOC.
H5_Heading5	A level-five heading. Not included in the TOC.
HRH_HelpRelatedHeading	A heading above a list of help topics that typically says "Related Topics" and is conditionalized to appear only in the online Help.
HRT_HelpRelatedTopics	Heading Only cross-references to related topics included in the online Help. The paragraphs are conditionalized to appear only in the online Help.
1. N0_Numbered	A numbered list.
a. NA_NumberedAlpha	An alphabetical list nested in a numbered list.
NOTE: NT0_Note **NOTE:** NT1_NoteLevel1 **NOTE:** NT2_NoteLevel2	A note related to the surrounding text but not strictly part of the flow. NT0 is flush left with body text. NT1 is indented .25 inches, and NT2 is indented .5 inches
➢ ST0_StepIntro	An introduction to a procedure.
TB0_TableBody	A paragraph of body text used only in tables.
TB1_TableBodyLevel1	A paragraph of body text indented .25 inches and used only in tables.
• TBU0_TableBulleted	A bulleted list in tables.
`TC0_TableCode`	A paragraph of code used only in tables.
`TC1_TableCodeLevel1`	A paragraph of code indented .25 inches and used only in tables.
TH_TableHeading	A heading cell in a table.
1. TN0_TableNumbered	A numbered list used only in tables.
NOTE: TNT0_TableNote	A note used only in tables.
NOTE: TNT1_TableNoteLevel1	A note indented .25 inches and used only in tables.

Use this tag…	To format …
TT_TableTitle	A title for tables. This format has no autonumber.
WARNING: TWN0_TableWarning	A warning used only in tables.
Question/Issue: WC_WritersComment	A comment or question for reviewers.
WARNING: WN0_Warning **WARNING:** WN1_WarningLevel1	A warning that is related to the surrounding text but isn't strictly part of the flow. WN1 is indented .25 inches
zzChapterNumber (sample not shown)	Don't use this format. It appears on master pages to format the autonumber next to the running header.
zzFooter	Don't use this format. It appears on master pages to format the footer text.
zzHeader	Don't use this format. It appears on master pages to format the running header text.

Character formats

The following table lists the character formats in the NICE FrameMaker templates and describes when to use them.

Use this tag…	To format …
`C0_Code`	Programming language keywords (such as `true`, `false`, and `int`), file names, and statements in body text.
CV_CodeVariable	Programming elements in italic to indicate that the formatted text is a placeholder.
Default ¶ Font'	Index marker text when you want to restore the default paragraph font. The tag must appear exactly like this in the index template's Character Format Catalog.
EB_EmphasisBold	Body text in bold for emphasis.
`EC_EmphasisCode`	Programming elements in bold and green for emphasis.
EI_EmphasisItalic	Book titles, words for emphasis, and new terms that require defining or explaining. Used with terms that appear in the glossary. Also used to format *See* and *See also* in index markers.
EMB_EmphasisMercedBlue	Figure callout text that you want to emphasize.
FIC_FigureCallout	Figure callout text.
GO_GuiOption	A UI element such as a button command, field name, or menu name in body text. (Leave tab names in the default font.)

Use this tag…	To format …
HL_HyperLink	A URL. Use only when you want to create a live link using a hyperlink marker.
`TCO_TableCode`	Programming language keywords in tables.
TEB_TableEmphasisBold	A UI element such as a button command, field name, or menu name in table cells. (Leave tab names in the default font.)
TGO_TableGuiOption	A UI element such as a button command, field name, or menu name in table cells. (Leave tab names in the default font.)
Do not use the following character formats in body text. They are used in autonumbers and cross-reference formats.	
zzChapterNumber	The autonumber format for AT_AppendixTitle and CT_ChapterTitle paragraphs.
zzChapterTitle	The Running H/F1 variable, used to display the chapter title on master pages.
zzEmphasisBoldBlue	The autonumber format for notes and warnings.
zzPageNum	The current page number variable on master pages.
zzStepintro	The autonumber format for StepIntro paragraphs.

Cross-reference formats

The following table lists each cross-reference format and shows an example of each one.

Use this tag…	To create this type of cross-reference…
Appendix	Appendix A, "Using the Templates,"
Appendix (end of sentence)	Appendix A, "Using the Templates."
Appendix (list item)	Appendix A, "Using the Templates"
Chapter	Chapter 1, "Working in FrameMaker,"
Chapter (end of sentence)	Chapter 1, "Working in FrameMaker."
Chapter (list item)	Chapter 1, "Working in FrameMaker"
Glossary Item	**Installing fonts**
Heading Only	"Paragraph formats"
Heading Only (end of sentence)	"Paragraph formats."
Heading & Page	"Paragraph formats" on page 157
On Page	on page 157
Page	page 157
Paragraph Number Only	A
Paragraph Text	Installing fonts
Paragraph Text (Plain)	Installing fonts

Use this tag…	To create this type of cross-reference…
See Heading & Page	See “Paragraph formats” on page 157.
Table All	Table , “TT_TableTitle,” on page 160
Table Number & Page	Table on page 167

Formatting examples

The following examples show paragraph format tags in blue and character format tags in red.

Cover formats

The following formats are in the `cover_SalesPM.fm` and `cover_ServicePM.fm` templates:

Copyright formats

The copyright text is an inset that FrameMaker automatically updates if the text changes. The following formats are in the `cover_SalesPM.fm` and `cover_ServicePM.fm` templates:

Copyright © 2012 NICE Systems Ltd. All rights reserved.

Information in this document is subject to change without notice and does not represent a commitment on the part of NICE Systems Ltd. The systems described in this document are furnished under a license agreement or nondisclosure agreement.

All information included in this document, such as text, graphics, photos, logos and images, is the exclusive property of NICE Systems Ltd. and protected by United States and international copyright laws.

Permission is granted to view and photocopy (or print) materials from this document for personal, non-commercial use only. Any other copying, distribution, retransmission or modification of the information in this document, whether in electronic or hard copy form, without the express prior written permission of NICE Systems Ltd., is strictly prohibited. In the event of any permitted copying, redistribution or publication of copyrighted material, no changes in, or deletion of, author attribution, trademark legend or copyright notice shall be made.

This product is covered by one or more of the US patents listed at www.nice.com/Patents

ACTIMIZE, Actimize logo, Alpha, Customer Feedback, eGlue Interact, FAST, FAST alpha Silver, Fortent, Fortent Logo, IEX, Insight from Interactions, Intent. Insight. Impact., Interaction Capture Unit, Know More, Risk Less, Last Message Replay, Mass Detection, Center, Mirra, My Universe, NICE, NICE Analyzer, NICE Inform, NICE Logo, NICE Perform, NICE Situator, NICE SmartCenter, NICE Storage Center, NICE Systems, NiceCall, NiceCall Focus, NiceLog, NiceTrack, NiceTrack IP Probe, NiceTrack Location Tracking, NiceTrack Mass Detection Center, NiceTrack Monitoring Center, NiceTrack Pattern Analyzer, NiceTrack Traffic Analysis, NiceVision, NiceVision Alto, NiceVision Analytics, NiceVision ControlCenter, NiceVision Digital, NiceVision Net, NiceVision NVSAT, NiceVision Pro, Open Situation Management, Playback Organizer, POWERCOACH, Scenario Replay, Searchspace, Syfact, Syfact Investigator, TotalView are trademarks and/or registered trademarks of NICE Systems Ltd.

Adobe® and Adobe Acrobat® are registered trademarks of Adobe Systems Incorporated in the United States.

Microsoft® and Microsoft Excel® are registered trademarks of Microsoft Corporation.

Highcharts® is a registered trademark of Highsoft Solutions AS.

Mule ESB® is a registered trademark of MuleSoft, Inc.

Apache Tomcat® and Tomcat® are registered trademarks of Apache Software Foundation.

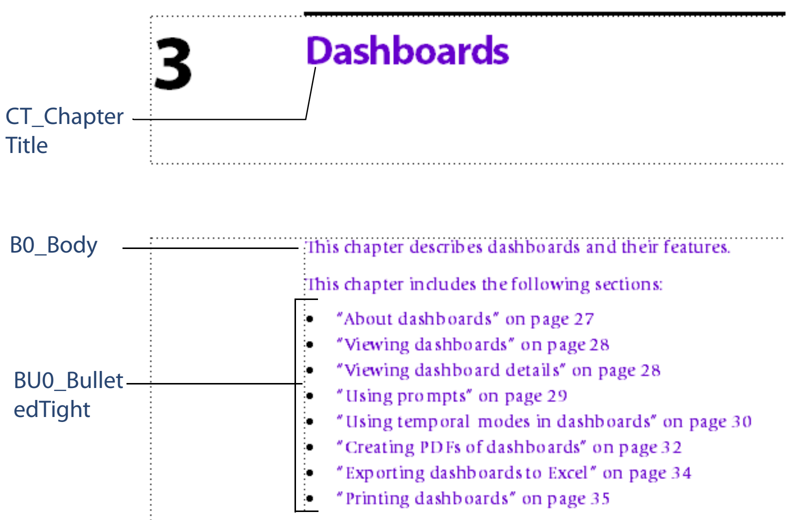
All other trademarks are the property of their respective owners.

Applications to register certain of these marks have been filed in certain countries, including Australia, Brazil, the European Union, Israel, Japan, Mexico, Argentina and the United States. Some of such registrations have matured to registrations.

Chapter formats

First page

The following graphic shows the formats used on the first page of a chapter:



Headings

Graphics

Table formats

Use the TB0_TableBody format for text in table cells. You can also use special formats within table cells, like notes and bulleted lists.

The chapter template contains the following table formats:

B0_Basic

TH_TableHeading	TH_TableHeading
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody

B1_BasicLevel1

TH_TableHeading	TH_TableHeading
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody

BW_BasicWide

TH_TableHeading	TH_TableHeading
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody

C_Columns

TH_TableHeading	TH_TableHeading
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody

FF0_FigureFrame

TH_TableHeading	TH_TableHeading
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody

FF1_FigureFrameLevel1

TH_TableHeading	TH_TableHeading
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody

LOT_LinedTitled

TT_TableTitle

TH_TableHeading	TH_TableHeading
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody

L0_Lined

TH_TableHeading	TH_TableHeading
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody

T0_Titled

TT_TableTitle

TH_TableHeading	TH_TableHeading
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody

T1_TitledLevel1

TT_TableTitle

TH_TableHeading	**TH_TableHeading**
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody
TB0_TableBody	TB0_TableBody

Index