

ecromedos 2.1 - Users' Manual

Tobias Koch

January 12, 2012

ecromedos.net

© 2009–2012 Tobias Koch

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled “GNU Free Documentation License”.

Contents

1. Introduction	6
1.1. What is ecromedos?	6
1.2. What's new in Version 2?	7
1.3. Additions in Version 2.1	7
1.4. Licensing	7
1.5. About this Document	7
1.6. Typographic Conventions	8
1.7. Getting Involved	8
1.8. Acknowledgements	8
2. Installation	9
2.1. Obtaining the Latest Version	9
2.2. Dependencies	9
2.3. Installation from Source	10
2.4. Installing Binary Packages	11
2.4.1. Debian/GNU and Ubuntu Linux	11
2.4.2. OpenSUSE Linux	11
3. General Usage Instructions	13
3.1. Beginning a New Document	13
3.1.1. Choosing a Document Class	13
3.1.2. Starting from a Template	13
3.2. Transforming Documents	14
3.2.1. Producing XHTML Output	14
3.2.2. Producing Postscript and PDF	15
3.3. Output Options	16
3.3.1. Specifying the Document's Language	16
3.3.2. Chunking Into Multiple Files	16
3.3.3. Chapter and Section Numbering	17
3.3.4. Generating a Table of Contents	17
3.3.5. Options Specific to Printable Output	17
4. Basic Document Editing	19
4.1. A Brief Introduction to XML	19
4.2. How to Structure Your Documents	20
4.3. Formatting Text	22
4.4. Controlling Hyphenation	22
4.5. Manually Inserting Line or Page Breaks	23
4.6. Working with Cross-References	23
4.7. Counters	24

4.8. Placing Marginal Notes and Footnotes	24
4.9. Inline and Block Quotes	25
4.10. Useful Pre-Defined Entities	25
5. Advanced Language Features	27
5.1. Using Lists	27
5.1.1. Bullet Lists and Ordered Lists	27
5.1.2. Definition Lists	28
5.2. Defining Tables	29
5.2.1. Basic Tables	29
5.2.2. Activating the Grid Rules	30
5.2.3. Coloring Table Cells	31
5.2.4. Text-Alignment in Table Cells	31
5.2.5. Rows Spanning Multiple Columns	31
5.2.6. Subtables	31
5.3. Embedding Graphics	32
5.4. About Floating Objects	33
5.5. Verbatim Text and Code Listings	33
5.6. Mathematic Formulas	34
5.6.1. Inline Math	34
5.6.2. Formulas as Block Elements	35
6. The Backmatter	36
6.1. The Appendices	36
6.2. Creating a Glossary	36
6.2.1. Making Definitions on the Spot	36
6.2.2. Automatic Generation of the Glossary	37
6.2.3. Creating the Glossary Manually	38
6.3. Keeping a Bibliography	38
6.4. Generating Keyword Indexes	39
6.4.1. Placing Index Markers	39
6.4.2. Automatic Index Generation	40
6.4.3. Manual Index Creation	41
7. Styling Your Documents	42
7.1. Editing the CSS-Stylesheet	42
7.1.1. Frontpage and Table of Contents	42
7.1.2. Text Body and Margin	43
7.1.3. Navigation Elements	43
7.1.4. Glossary and Index	44
7.2. Changing the Look of L ^A T _E X-based Output	45
A. GNU General Public License	48
B. GNU Free Documentation License	54
Bibliography	62

Language Elements

63

1. Introduction

If you are the type of person whom the typical word processor drives out of their mind. If your blood pressure goes up, when a piece of software tries to be smart about every single key press you make, even interpreting the backspace key in a myriad of ways that you never expected. Or if you find yourself spending more time on reformatting your documents than doing actual writing, then *ecromedos* might be just for you.

This is about a piece of software that can change your life as a technical writer and restore your belief in the power of the backspace key, for good. Making use of modern technologies, but building on proven concepts that stem from the early days of computing, *ecromedos* is a text processing system of a different kind. If you are ready to break with old habits and willing to learn something new, then I hope you will find that *ecromedos* can render you more productive by letting you focus on writing, and nothing else.

1.1. What is *ecromedos*?

ecromedos is an integrated solution for XML-based publishing in print and on the Web. It is primarily targeted at, but not limited to, the creation of technical documentation in the field of Computer Science. Documents are written in a semantic markup language and converted to representational formats with a special processing toolchain. Currently, *ecromedos* supports the target formats XHTML and \LaTeX , where the latter can be compiled into high-quality PostScript or PDF by use of the \TeX ⁽¹⁾ typesetting system.

The *ecromedos Markup Language* (ECML) is modelled closely after HTML with some ideas and additional elements borrowed from \LaTeX . It allows you to compose comprehensive, well-structured documents from a comparably small set of language elements. Users who are already familiar with HTML or document markup languages like DocBook will find learning ECML particularly easy.

The *ecromedos Document Processor* (eDP) uses XSL Transformations (XSLT) to convert source documents from ECML to the supported output formats. In addition, the eDP contains a specialized pre-processor, which operates directly on the XML-DOM tree of a document and carries out additional preparational tasks that could not be implemented with bare XSLT, such as the conversion of images to processable formats or on-the-fly syntax-highlighting of source code listings.

ecromedos is well-suited for environments where many writers collaborate on many documents. Since documents are written in a semantic markup language, writers do not have to worry much about following a style guide. There is very little coordination required in achieving a consistent layout and look over an entire range of documents, because the layouting and formatting is done automatically by the eDP and assorted tools during the conversion to representational formats.

⁽¹⁾More information on \TeX and \LaTeX can be found in the *Comprehensive \TeX Archive Network* at <http://www.ctan.org>.

1.2. What's new in Version 2?

Starting with version 2, ecromedos can be considered a full-blown document authoring system. Beside some language clean-ups in the DTD, a couple of new features were added that make the new ecromedos much more versatile than version 1.

The most notable deficits of version 1, namely the lack of support for glossary and keyword indexes, have been remedied. The new DTD lets you make definitions "on the spot", and place index markers as you are writing. The document pre-processor collects all terminological definitions and index markers while parsing the document and automatically builds the corresponding indexes.

The table model has been enhanced to allow for table cells to be split into subtables. This compensates for the lack of support for cells spanning multiple rows. A new pre-processor plugin auto-detects the number of vertical frame rules in a table, making it unnecessary to provide explicit hints in the column specification of tables.

Table cells and list items may now contain arbitrary combinations of block elements and are no longer restricted to carrying plain text only. This opens up new possibilities for the conversion of complex HTML documents to ecromedos Markup Language.

Last but not least, some major effort has gone into removing formatting information, such as CSS styles, from the XSL stylesheets wherever possible and providing central styles definition files. Thus, simple customizations no longer require digging into the XSL stylesheets.

1.3. Additions in Version 2.1

Starting from version 2.1, Pygments⁽²⁾ is used for syntax highlighting. It replaces the old built-in syntax highlighter. Pygments provides support for lexing a vast amount of programming languages and scripts and is a very mature and robust product.

1.4. Licensing

ecromedos is free (as in speech) software, licensed under the *GNU General Public License, version 2* [GPL2]. You are allowed to take the source code and modify it as you please, as long as you publish your changes under the same license. You *must not* sell ecromedos or a derived work or embed it into a closed-source proprietary application or library. A verbatim copy of the license text can be found in appendix A.

1.5. About this Document

This document describes the installation and usage of ecromedos and the scope of the ecromedos Markup Language. After reading this manual, you will be able to write feature-rich documents and translate them to representational on-screen or printable formats with the ecromedos Document Processor (eDP).

- In chapter 2, you will find instructions for installing ecromedos from source or binary packages.

⁽²⁾<http://pygments.org>

- Chapter 3 teaches you the basics of starting a new document and transforming documents to representational formats with the eDP.
- After reading chapter 4, you will know how to write simple documents, do basic text-formatting and work with cross-references and counters.
- Chapter 5 introduces some of the more advanced language constructs, teaching you how to use lists, define tables, set mathematic formulas, or embed images into your documents.
- Chapter 6 covers everything related to the backmatter of a document, including how to keep a bibliography, how to cite from it, and how to generate glossary and keywords indexes.
- Chapter 7 explains how to create your own style definitions to change the appearance of rendered output.

1.6. Typographic Conventions

Throughout this document the following typographic conventions apply:

- ECML language elements mentioned in the text will be written in bold-face typewriter letters.
- The names of filesystem folders and links (URLs) to external resources will be set in typewriter letters.
- Examples of command line text input will be set in typewriter letters. The character sequence `#>` represents a shell prompt.
- Program listings will be set in typewriter letters on a gray background.
- Italicized text is used in listings for placeholders that need to be substituted by the user.

1.7. Getting Involved

ecromedos version 2 is a big step forward from version 1, but there still is a lot to do. There are many ways in which you can contribute. You can report bugs and help squash them, you can write documentation, and you can submit patches for new features. If you have any suggestions or would like to participate in future development, please write to development@ecromedos.net.

1.8. Acknowledgements

I would like to thank Prof. Dr. Heinz-Erich Erbs from the [University of Applied Sciences in Darmstadt](#), Germany, for his support and advice when I started work on ecromedos as an undergraduate project.

A big “thank you” goes out to Daniel Veillard for writing a phantastic set of libraries for processing XML, to Guido van Rossum and the Python Software Foundation for a beautiful scripting language, to Donald Knuth for the coolest typesetting program around, and to André Simon for a lecture on how to write a syntax-highlighter⁽³⁾, that I got for free by looking at his code.

Thanks to everybody working on Free Software, your stuff is a lot of fun!

⁽³⁾Newer versions of ecromedos use the [Pygments](#) syntax highlighter. Versions before 2.1 came with a built-in syntax highlighter

2. Installation

This chapter covers the installation of ecromedos from source and binary packages. You should be able to run ecromedos on any system on which the dependencies listed below are available.

2.1. Obtaining the Latest Version

The latest version of the software can be obtained from the project's website at <http://www.ecromedos.net>. You will find a tarball for manual installation and binary packages for selected platforms.

2.2. Dependencies

For proper operation, ecromedos depends on the following third-party software offerings:

Python

A powerful scripting language. Python is available for all popular operating systems and can be obtained from the website of the Python Software Foundation at <http://www.python.org>. You will need version 2.4 or later to run ecromedos. Please note that ecromedos is not yet ready for Python 3.

libxml and libxslt

Two super fast and stable libraries for processing XML, developed by Daniel Veillard for the GNOME project. Both libraries offer bindings to Python. The project homepage is at <http://www.xmlsoft.org>.

Imagemagick

A set of tools for image manipulation. ecromedos uses Imagemagick for resizing bitmap images and to do format conversions. You can download the software from <http://www.imagemagick.org>.

T_EX

If you want to produce Postscript or PDF, you need an installation of the T_EX typesetting system. It is recommended that you use T_EX Live 2009 or later, available from <http://www.tug.org/texlive/>.

dvipng

A utility to convert DVI files to PNG or GIF graphics. ecromedos needs this to convert mathematic formulas to bitmap graphics when generating XHTML output. dvipng is available from <http://www.nongnu.org/dvipng/>, it may also be included in your T_EX distribution.

Pygments

Pygments is a powerful syntax highlighter that can lex a vast amount of programming languages

and scripts. ecromedos uses Pygments for automatic syntax highlighting of code samples. Find out more about Pygments at <http://pygments.org>.

2.3. Installation from Source

First, double check that you have installed the dependencies listed in section 2.2.

Step 1: Unpacking the Tarball

ecromedos can be installed anywhere in the file system. It is recommended, however, that you place it in `/opt`. In order to do so, open a shell and become superuser root by typing

```
#> su -
```

Then change your working directory to `/opt`

```
#> cd /opt
```

and unpack the tarball. The following example assumes that the tarball resides in the home directory of system user tobias:

```
#> tar -xvzf /home/tobias/ecromedos-x.y.z.tar.gz
```

Step 2: Adding the Executable to the System Path

Now change to `/usr/local/bin` and create a symbolic link to the main executable:

```
#> cd /usr/local/bin
#> ln -s /opt/ecromedos-x.y.z/bin/ecromedos
```

By placing this link in the system path, you will be able to call ecromedos by name without having to enter the canonical path to the executable.

Step 3: Pre-compiling the Source Files

Since you would typically install ecromedos in a location, to which normal system users don't have write access, you should pre-compile the Python source files to bytecode, so that users can benefit from much-improved application start-up times. To do so, change *into* the installation directory and issue the following command:

```
#> find . -name "*.py" -exec \
    python -c "import py_compile; py_compile.compile('{}')" \;
```

Step 4: Adapting the Configuration

Having completed the installation procedure described above, you may now edit the configuration file `ecmds.conf`, which is located in the subdirectory `etc` below the installation folder. ecromedos is designed

to be relocatable, so in many cases it should just work out of the box. However, the configuration variables holding pointers to third-party software that ecromedos relies on may have to be corrected. In particular, check all variables ending in `_bin` or `_dir`.

The value of a configuration variable can be reused in arguments to other variables by prefixing the name of the variable to be referenced with a dollar sign. For instance, if `base_dir` is set to `/opt/ecromedos-x.y.z/`, then you can set `lib_dir` to `/opt/ecromedos-x.y.z/lib` by simply writing `lib_dir = $base_dir/lib`.

ecromedos 2 introduces the automatic variable `$install_dir`, which is set implicitly and points to the installation directory.

2.4. Installing Binary Packages

Packages for Debian, Ubuntu and OpenSUSE are provided. Since ecromedos is written in Python, and thus does not need to be compiled or linked against system libraries, it should run on any version of these distros, as long as the dependencies listed in section 2.2 are available.

2.4.1. Debian/GNU and Ubuntu Linux

If you are running Debian/GNU or Ubuntu Linux, you can install ecromedos via apt. Just insert the following entry into `/etc/apt/sources.list`:

```
deb http://ecromedos.net/packages/debian debian contrib
```

Then update the package cache and install ecromedos by saying

```
#> apt-get update
#> apt-get install ecromedos
```

The documentation is contained in a separate package `ecromedos-doc`.

2.4.2. OpenSUSE Linux

To comfortably install ecromedos on OpenSUSE, add the following repository URL to your repository sources:

```
http://ecromedos.net/packages/opensuse
```

You can either do this from YaST or on the command line using `zypper` like this:

```
#> zypper addrepo -c -f http://ecromedos.net/packages/opensuse/ ecromedos2
#> zypper install ecromedos
```

And if you also want to install the documentation:

```
#> zypper install ecromedos-doc
```

3. General Usage Instructions

This chapter covers the basics of starting a new document and using the `ecromedos` command line tool to convert documents to representational formats, also touching on the available output options that allow basic control over how documents are rendered.

3.1. Beginning a New Document

Starting a new document is easy and boils down to

1. choosing a document class and
2. generating a template, from which to continue editing.

3.1.1. Choosing a Document Class

When starting a new document, you first have to choose a *document class* fitting the type of document you intend to write. Document classes are abstract definitions of typical real-world document types, such as 'letter', 'book', or 'newspaper article'.

The `ecromedos` Markup Language (ECML) provides the three document classes `article`, `book` and `report`. The primary difference between these is how many section levels they provide and how they are rendered when generating printable output.

article

The article class is intended for short documents without the need for a glossary or keyword index. On paper, articles are rendered single-sided with an in-page title. Articles offer three section levels through the sectioning elements `section`, `subsection` and `subsubsection`.

book

The book class is for large documents that may contain a glossary and keyword indexes. In print, documents are rendered double-sided with a separate title page. This class introduces the top-level sectioning element `chapter`. Chapters start on odd-numbered pages and may be further grouped into `parts`.

report

A report is essentially the same as a book, but documents are rendered single-sided when generating printable output.

3.1.2. Starting from a Template

When you have decided on which document class you want to use, the recommended way of starting your document is to generate a bare document template. In order to do so, open a terminal window and issue the following command:

```
#> ecromedos -n class > index.xml
```

Where *class* should be one of **article**, **book**, or **report**. When you open the file `index.xml` in your editor, you should see something similar to the following listing.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report SYSTEM
  "http://www.ecromedos.net/dtd/2.0/ecromedos.dtd">
<report lang="en_US" secnumdepth="3" secsplitdepth="1">
  <head>
    <subject>Subject</subject>
    <title>Title</title>
    <author>Author</author>
    <date>Date</date>
    <publisher>Publisher</publisher>
  </head>
  <make-toc depth="3" lof="no" lot="no" lol="no"/>
  ...
  <chapter>
    <title>Chapter Title</title>
    <p>
      First paragraph in first chapter...
    </p>
  </chapter>
  ...
</report>
```

Even if you are not yet familiar with the ecromedos Markup Language, you should be able to interpret the general structure of the markup. For now, you don't need to know the details. You will dive deeper into document writing in chapter 4.

3.2. Transforming Documents

In order to convert your documents from ecromedos Markup Language to one of the supported output formats, you have to call the ecromedos Document Processor from the command line. The exact procedures are explained in the following subsections.

3.2.1. Producing XHTML Output

XHTML is the default output format, unless you modified the corresponding setting in the configuration file. In the default installation, to convert a document from ECML to XHTML, you simply have to issue the following command:

```
#> ecromedos document
```

where *document* is the name of an ECML input file. The output files will be placed into the working directory of the ecromedos command. If you don't want the output in the same directory as the source files, simply create an empty folder, change into it, and call ecromedos from there:

```
#> mkdir spool
#> cd spool
#> ecromedos ../document
```

ecromedos will copy all resources required for viewing the document to the output folder and, in the case of HTML output, adjust the references inside the transformed document accordingly. This means that even though you can share images, such as logos, between source documents, each representational instance of a document will be completely self-contained.

After transformation, load the file `index.html` into your web browser, to view the results.

3.2.2. Producing Postscript and PDF

If you want to produce a Postscript or PDF version of your document, you will have to generate \LaTeX output first. Use the `-f` command line switch to tell ecromedos the desired target format:

```
#> ecromedos -f latex document
```

Depending on your setting of `secsplitdepth` (see section 3.3.2), you will obtain one or more output files with the extension *tex*. The main \TeX -file will carry the name of the document class you are using. To compile the \LaTeX sources of your document, invoke the \LaTeX compiler like this:

```
#> latex class.tex
```

You may have to call \LaTeX two or three times until all cross-references are resolved and the table of contents is completely built. The result will be a file with the extension *dvi*. You can use the following commands to convert the DVI file to PostScript and PDF:

```
#> dvips -Ppdf class.dvi
#> ps2pdf class.ps
```

The first command will generate Postscript output from the DVI file and the second command will turn the Postscript into PDF. Starting with ecromedos 2.0, you can take advantage of the fact that \TeX can produce PDF directly through the *pdftex* driver. So instead of taking the detour over an intermediate Postscript file, you can instruct ecromedos to produce output that can be compiled with the `pdflatex` command:

```
#> ecromedos -f pdflatex document
#> pdflatex class.tex
```

In addition, there is some preliminary support for generating output that can be processed with the Unicode-aware X_YTeX using the `xelatex` command:

```
#> ecromedos -f xelatex document
#> xelatex class.tex
```

And again, remember that you may have to call L^AT_EX multiple times before the document is rendered complete.

3.3. Output Options

Even though we are dealing with semantic markup, there are some decisions about the presentation of a document that are left to the author, such as whether the document should have a table of contents or not, whether sections are to be numbered and if so, down to which section level, and so on. The language elements and element attributes that are described below, give you some limited control over these presentational aspects.

3.3.1. Specifying the Document's Language

By setting the `lang` attribute on the document's root element you can select the language to be used for automatic titles, i.e. section titles that are generated by the document processor automatically, such as "table of contents" or "bibliography". When generating L^AT_EX output, this also activates the hyphenation patterns for the specified language.

In ecromedos version 1 you could simply supply the english name of the desired language such as `german` or `english`. Starting with version 2, you have to supply an ISO locale identifier such as `en_US`. This is to better take into consideration regional differences. For example, there are countries that speak the same language but use different scripts.

Currently, ecromedos supports the following language/territory combinations:

Language	Territory	Identifier
English	Canada	en_CA
	Great Britain	en_GB
	New Zealand	en_NZ
	USA	en_US
German	Austria	de_AU
	Germany	de_DE
	Switzerland	de_CH

3.3.2. Chunking Into Multiple Files

Set the `secsplitdepth` attribute on the root element of your document to an integer value between 0 and 5, in order to control down to which section level the document will be chunked into individual files. This is especially useful when generating HTML output. Splitting up large HTML documents into multiple files will improve the user experience, because the document will be easier to navigate,

individual parts will load more quickly, and the user's browser doesn't have to keep the entire document in memory, at once.

Here is an example on how it works. When you write a book or report without making use of the `part` element, setting `secsplitdepth` to 1 will result in each chapter being written to a separate file. The level count always starts at zero. So if, one day, you decide to group the chapters into parts, leaving `secsplitdepth` at 1 means that the document is now split at the parts' level. If you still want each chapter to go to a separate file, you have to increment `secsplitdepth` to two. If you set `secsplitdepth` to zero, the entire document will be rendered into a single file.

When generating HTML, the ecromedos Document Processor will take care of linking the document together via a navigation menu that will appear at the top and bottom of each output file, providing for a seamless reading experience.

3.3.3. Chapter and Section Numbering

Set the `secnumdepth` attribute on the document's root element to an integer value between 0 and 5, in order to control down to which section level sections are to be numbered. Setting it to zero will turn section numbers off completely.

3.3.4. Generating a Table of Contents

In previous versions of ecromedos the generation of the table of contents (TOC) was controlled by setting the root element's `tocdepth` attribute appropriately. Starting with version 2.0, there is a new language element `make-toc` for that purpose, which should be inserted right after the document header.

By setting the element's `depth` attribute to an integer value between 0 and 5, you determine the deepest section level that will be included in the TOC. In addition, you may set either of the attributes `lof`, `lot`, `lol` to yes or no, to toggle whether the TOC should contain a "List of Figures", "List of Tables" or "List of Listings", respectively.

Section Overviews

There is some preliminary support for per-section overviews. To generate a mini table of contents for a section, simply put a `make-overview` element right at the beginning of the section. There is one caveat however: \LaTeX only supports section overviews for the base sections, i.e. in the book and report classes you can have section overviews for parts and chapters but not for sections and subsections. For HTML this limitation does not exist.

3.3.5. Options Specific to Printable Output

The attributes `papersize`, `bcor` and `div`, which may be set on the document's root element, are passed through to \LaTeX and influence how your document is rendered when producing Postscript and PDF.

The purpose of the `papersize` attribute should be pretty obvious. Supported values are `legalpaper`, `letterpaper`, `executivepaper`, `aXpaper`, `bXpaper`, `cXpaper`, `dXpaper`. The default is `a4paper` which is the standard office paper format used in Germany. You can activate landscreen mode by adding the keyword `landscape` separated with a comma.

The `bcor` attribute lets you specify a binding correction. That is the amount in centimeters (cm) or points (pt) by which the text body should be indented to make up for margin space lost when binding the document. For example write `bcor="1.5cm"` to get a binding correction of 1.5 centimeters.

The `div` attribute indirectly controls the dimensions of the text body. Its argument is passed through to the \LaTeX macro package KOMA-Script which is responsible for layouting the document. KOMA-Script tries to automatically determine the optimal dimensions for the text body by applying a set of typographic rules. To this end, the page is divided into $div \times div$ rectangles of equal size, which serve as the basic units for splitting the page into margins and text body. The greater you choose `div`, the larger the text area will be. Try values between eight and 16.

Use the `parskip` attribute to specify the amount of horizontal space to insert in between paragraphs of text. You have the choice between `full` for a full line, `half` for half a line (default) or `off` to have no skip in between paragraphs. In the last case, the first line of each paragraph will be indented.

4. Basic Document Editing

This chapter starts with a crash course in XML and then continues teaching you the essentials of writing documents in ecomedos Markup Language.

4.1. A Brief Introduction to XML

The *Extensible Markup Language* is not a data description language in itself. Rather, it defines a syntax that allows you to design your own customized markup languages for arbitrary data models. Take a look at the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<cocktail alcoholic="yes">
  <name>Pina Colada</name>
  <ingredient>
    <name>rum</name>
    <amount unit="oz">3</amount>
  </ingredient>
  <ingredient>
    <name>coconut milk</name>
    <amount unit="tbsp">3</amount>
  </ingredient>
  <ingredient>
    <name>pineapple</name>
    <amount unit="tbsp">3</amount>
  </ingredient>
  <ingredient>
    <name>ice</name>
    <amount unit="cup">2</amount>
  </ingredient>
</cocktail>
```

What you see is a complete XML document describing the popular Pina Colada cocktail. On the first line you see the XML declaration. It specifies the XML version and the character encoding of the document. In general, you should use one of the unicode character encodings UTF-8 or UTF-16, since any standards-conforming XML parser is required to be able to read these. What follows is the root element `cocktail` which in turn contains a `name` element, telling us the name of the cocktail, followed by the various ingredients that you need to make a Pina Colada.

The textual elements delimiting the beginning and end of an XML element are called *tags*. An opening tag has the form `<element>` and the corresponding closing tag is written `</element>`. XML elements must be properly nested and there can be only one root element. Thus, you can picture the logical structure of an XML document as a tree. A single, bare element of this tree is called a *node*. Its direct

descendants are called its *children* and the node from which it originates is called the *parent*. Nodes may carry additional attributes. Our cocktail, for instance, has the attribute `alcoholic` which in this case is set appropriately to `yes`.

The names of elements and attributes can be chosen arbitrarily to represent a given data model. In this example we tried to model a drink but you might just as well define a set of tags to describe the parts of an automotive vehicle. When working with ecromedos, you will be using XML-based markup to describe the logical structure of standard text documents.

4.2. How to Structure Your Documents

In general, you should start new documents from a document template, as described in section 3.1.2. An initial template produced that way will contain a complete skeleton for a document of the chosen document class. For a book or report the overall structure of a document can be represented by the following tree:

```
report
|-head
|-legal
|-make-toc
|-preface
.
.
|-chapter
| |-title
| |-blockelements
| |-section
| . |-title
| . |-blockelements
|   |-subsection
|     . |-title
|     . |-blockelements
|       |-subsubsection
|         . |-title
|         . `~blockelements
.
.
|-appendix
| |-title
| |-blockelements
| |-section
| .
| .
.
.
|-glossary
|-biblio
|-index
.
```

This tree is greatly simplified and incomplete, because naturally any type of sectioning element, with the exception of **subsubsection**, can contain multiple subordinate sections. When using the article class, the **preface** element is replaced by **abstract** and the elements **legal**, **glossary** and **index** are not available. Furthermore, the top-level sectioning element in an article is the **section**.

Block elements may be figures, equations or tables; or simply paragraphs of text, which are set with the **p** element.

Document Header And Legal Section

The very first child element of a document's root element is always the document header section, which has the following structure:

```
<head>
  <subject>Special Subject, e.g. Ph.D. Thesis</subject>
  <title>Document Main Title</title>
  <subtitle>Subtitle</title>
  <author>Author 1</author>
  <author>Author 2</author>
  ...
  <date>Date of Publication</date>
  <publisher>Name of Publisher</publisher>
</head>
```

Contrary to HTML, where the order of the header elements can be arbitrary, in ECML the order is fixed. In books and reports the header can be followed by an optional **legal** section, which consists of paragraphs of text and which is meant to hold copyright information and related stuff. The legal section should generally fit on one single page of whatever paper format you have chosen for printable output.

The preface Element

In books and reports you may use the **preface** element to set an arbitrary number of introductory sections right after the document header and the table of contents. The title of a preface will not be numbered and it will not appear in the table of contents when generating printable output. A preface may contain paragraphs of text, as well as other block elements, such as figures and tables. It must not contain any deeper sections except for minisections. If you feel that you need to divide your preface, you should consider making it a chapter.

Minisections

A **minisection** is a special kind of section, that can occur anywhere in the section hierarchy. The title of a minisection will be set in smaller letters than any regular section title, it will not be numbered and also not listed in the table of contents.

Paragraphs

A paragraph is set with the **p** element and is the simplest block element available, containing only formatted text and inline elements. Paragraphs can also optionally have a **title**, which will be set inline. See section 4.7 for an example of how this can be useful.

4.3. Formatting Text

From your word processor you may be used to being able to emphasize text by setting it in bold or italic letters or by underlining it. With ecromedos you can achieve this by enclosing the span of text to be formatted inside the tags `b` for bold print, `i` for italics or `u` for underlining. You may also combine these arbitrarily. In addition, you may use the `tt` tag to make text appear in typewriter letters, which is useful for setting, for example, internet addresses.

Table 4.1: Using text-formatting elements

Markup	Resulting Output
<code><u>underlined text</u></code>	<u>underlined text</u>
<code><i>text in italics</i></code>	<i>text in italics</i>
<code>bold-faced letters</code>	bold-faced letters
<code><i>bold face and italics</i></code>	<i>bold face and italics</i>
<code><tt>text in typewriter letters</tt></code>	text in typewriter letters
Super <code><sup>script</sup></code>	Super ^{script}
Sub <code><sub>script</sub></code>	Sub _{script}
<code><xx-small>text in XXS</xx-small></code>	text in XXS
<code><x-small>text in XS</x-small></code>	text in XS
<code><small>small letters</small></code>	small letters
<code><medium>regular size</medium></code>	regular size
<code><large>large letters</large></code>	large letters
<code><x-large>text in XL</x-large></code>	text in XL
<code><xx-large>text in XXL</xx-large></code>	text in XXL
<code><color rgb="#880000">text in red letters</color></code>	text in red letters

For the sake of completeness, there are the seven elements `xx-small`, `x-small`, `small`, `medium`, `large`, `x-large` and `xx-large` that let you control the font size. However, there should hardly ever be a reason to change the font size explicitly. Use the elements `sup` and `sub` in order to set text in super or subscript.

You can color text with the `color` element as shown in table 4.1. The `rgb` attribute expects a color value in CSS-style hexadecimal notation.

You may have noticed that formatting text is one of the few areas where the ecromedos Markup Language (ECML) isn't purely semantic. If that bothers you, just pretend `i` stood for *emphasis* and discard the other elements. Alternatively, you could use these features to develop your own, specialized markup language on top of ECML, which is very much encouraged!

4.4. Controlling Hyphenation

In printable output, text is set justified over the entire width of the page's text area. In order to avoid large gaps of white space between words, \LaTeX uses a clever algorithm and language-specific patterns to hyphenate words automatically. However, sometimes the hyphenation algorithm fails and in rare cases it cannot hyphenate certain words, at all. You can provide hints, telling \LaTeX where a word may be broken up, by inserting `y` tags in the right spots. For example, in order to tell \LaTeX that it may split the

word “bibliography” only in between *biblio* and *graphy*, you would write `biblio<y/>graphy` in your markup.

4.5. Manually Inserting Line or Page Breaks

In general, you should not have to worry about where a line breaks or where a new page begins, because it is the job of the formatting engine (i.e. \LaTeX or your web browser) to take care of this. In rare cases, however, you may have to intervene manually. You can use the `br` element to break the current line or `pagebreak` to start a new page. You should *not* use multiple line or page breaks in a row.

When you need to *prevent* linebreaks in certain places, you can either use the non-breaking space (see section 4.10) or protect the specific strip of text with the `noBR` tag. For example, a title or academic degree should not be separated from the name that follows it. Consequently, you should write `Dr. Pepper` or `<noBR>Dr. Pepper</noBR>` to prevent the formatting engine from possibly breaking the line right before Pepper.

4.6. Working with Cross-References

Sometimes you will want to refer to another section in your manuscript, i.e. you may write something like, “[...] you will find out more about this on page XYZ”. However, at the time of writing your markup, you cannot tell on which page the section you are referring to will actually be printed. The solution is to label the locations you wish to reference and let ecromedos take care of filling in the correct number whenever it encounters a reference to a label in your document⁽¹⁾.

The syntax for the definition of cross-references has changed slightly in ecromedos version 2. To label a certain spot in the text, use the `label` tag. This tag has a single, mandatory `id` attribute. This must be a unique identifier among *all* elements that carry an `id` attribute. Take a look at the following example:

```
<chapter>
  <title>The Show about Nothing</title>
  <p>
    Seinfeld<label id="seinfeld"> is the best
    sitcom of all times.
  </p>
</chapter>
```

You can now use the `ref` element to obtain the section number and `pageref` to get the page number like this:

```
<chapter>
  <title>About Myself</title>
  <p>
```

⁽¹⁾ Depending on the target format, ecromedos may actually delegate the task of filling in cross-references to the formatting subsystem, such as is the case for \LaTeX output.

```

I really enjoy watching Seinfeld. You can read more
about Seinfeld in section <ref idref="seinfeld"/> on
page <pageref idref="seinfeld"/>.
</p>
</chapter>

```

The `ref` and `pageref` elements can also point to any other object with an `id` attribute, such as a `figure` or a numbered `equation`. In that case `ref` will resolve to the corresponding object counter instead of the section counter.

Hyperlinks

You can insert hyperlinks into your document with the `link` element, which has a single, mandatory attribute `url`, which is used in exactly the same way as the `href` attribute on HTML anchors:

```

Click this <link url="mailto:bobburnquist@example.com">link</link> to
send a mail to a non-existent e-mail address or visit
<link url="http://www.shredordie.com/">shredordie.com</link> for some
cool skate videos.

```

4.7. Counters

You can create new object counters with the `counter` element. For example, you may decide to create an “example” environment with its own object counter. An instance of such an “example” might look like this:

```

<p>
  <title>Example <counter group="example"
    simple="no" id="ex:counterhowto"/>:</title>
  <i>
    This is an example on how to use the <tt><b>counter</b></tt> element ...
  </i>
</p>

```

By giving the counter an `id`, you can cross-reference the counter using the `ref` and `pageref` elements (see section 4.6). If you set the optional `simple` attribute to `yes`, the section count will be omitted. In the rare event that you need to start counting from zero, set the optional `base` attribute to 0.

4.8. Placing Marginal Notes and Footnotes

Marginal notes can be placed with the `marginal` tag. And if you have Javascript turned on, they will even be displayed correctly in HTML output. Try this example:

```
<p>
  In this episode<marginal>The Summer of George</marginal>,
  George finally loses his job at the Yankee Stadium but
  gets an extra three months' pay-off.
</p>
```

Please note that \LaTeX does not allow setting marginals inside table cells. For HTML output this limitation does not exist. Footnotes are placed in the same fashion as marginals by use of the `footnote` tag. They do work from inside tables without restrictions.

4.9. Inline and Block Quotes

Unless you are setting your text in typewriter letters, you will not be able to enter the correct quotation marks for your language directly with your keyboard. You could use XML character entities to access the glyphs, but that is tedious. Instead you should use the tags `q` and `qq` for single and double quoting, respectively.

When quoting large portions of text, consider using the `blockquote` element, which acts as a block element and may contain multiple paragraphs of text. Block quotes will be indented left and right to set them off from the rest of the text.

4.10. Useful Pre-Defined Entities

ecromedos defines a small set of entities that may come in handy occasionally. Table 4.2 shows the available entity names and what they stand for. The zero-width space is particularly useful for making long path names or Internet addresses break across lines without introducing hyphens or spaces.

Table 4.2: Pre-Defined Entities

Entity	Resolves to
<code>&tex;</code>	\TeX
<code>&latex;</code>	\LaTeX
<code>&xetex;</code>	$X_{\text{E}}\TeX$
<code>&xelatex;</code>	$X_{\text{E}}\LaTeX$
<code>&nbsp;</code>	The non-breaking space
<code>&zwsp;</code>	The zero-width space
<code>&endash;</code>	–
<code>&emdash;</code>	—
<code>&dots;</code>	...
<code>&check;</code>	✓
<code>&cross;</code>	✗

For direct access to these entities, you must include the following document type declaration at the top of your document:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE report SYSTEM "http://www.ecromedos.net/dtd/2.0/ecromedos.dtd">
```

If you start your documents by generating a template, as described in section 3.1.2, the document type declaration will already be in place. Entities can also be accessed by name, through the `entity` element, without including the document type declaration. For example, you can insert an em-dash into the text by writing `<entity name="emdash"/>`.

5. Advanced Language Features

What you have learned up until now won't get you far, if you intend to write anything but plain prose. This chapter introduces some of the more advanced language constructs, that let you spice up your documents with illustrations, tables, structured lists, program listings and mathematic formulas.

5.1. Using Lists

ecromedos knows three types of lists:

- ordered lists,
- unordered lists (a.k.a bullet lists),
- and definition lists.

5.1.1. Bullet Lists and Ordered Lists

Bullet lists are set with the `ul` element and ordered lists with the `ol` element. List items are enclosed inside `li` tags. Both list types may be nested within each other up to four levels deep. Starting with ecromedos version 2, list items may contain arbitrary block elements. But please note that a block element inside a list *must not* bear a caption. The following listing shows an example of a nested list structure:

```
<ol>
  <li>
    <p>First paragraph in first list item</p>
    <p>Second paragraph in first list item</p>
  </li>
  <li>
    <p>Second list item</p>
    <ul>
      <li>
        <p>First subitem of second list item</p>
      </li>
      <li>
        <p>Second subitem of second list item</p>
      </li>
    </ul>
  </li>
  <li>
    <p>Third item in outer list</p>
    <ol type="i">
```

```

    <li>
      <p>Item one in ordered sublist</p>
    </li>
    <li>
      <p>Item two in ordered sublist</p>
    </li>
    <li>
      <p>Item three in ordered sublist</p>
    </li>
  </ol>
</li>
</ol>

```

Ordered lists at different nesting levels will receive different enumeration marks, such as arabic numbers, latin letters, or roman numerals, to reflect their position in the list hierarchy. The type of enumeration mark at a given level is selected automatically, but may be overridden by setting the list's `type` attribute as follows:

```

<ol type="1"> for arabic numbers (1, 2, 3, ...)
<ol type="i"> for roman numerals in lowercase (i, ii, iii, iv, ...)
<ol type="I"> for roman numerals in uppercase (I, II, III, IV, ...)
<ol type="a"> for latin letters in lowercase (a, b, c, ...)
<ol type="A"> for latin letters in uppercase (A, B, C, ...)

```

5.1.2. Definition Lists

Definition lists are set with the `dl` element. An item in a definition list has two components: a term or expression to be defined and its respective definition. Take a look at the following example:

```

<dl>
  <dt>ecromedos</dt>
  <dd>
    A document publication system that allows generating
    different target formats from one document source.
  </dd>
  <dt>ECML</dt>
  <dd>
    The ecromedos Markup Language is an XML based markup
    language for describing the logical structure of
    standard text documents.
  </dd>
</dl>

```

While the `dt` element may contain only simple text and text-formatting elements, the `dd` element may contain arbitrary sequences of block elements.

5.2. Defining Tables

Starting with version 2, ecromedos features a complete table model with table captions, cells that can span multiple columns, nested subtables and minute control over the visibility of the table grid. The language elements for setting tables were largely borrowed from HTML. However, there are some subtle differences between the HTML and the ECML table model, which will become apparent in the course of this section.

5.2.1. Basic Tables

Tables are likely the most complicated part of ECML. But once you know the ins and outs of the ECML table model, you will appreciate the ease with which you can create good-looking tables in your documents. To get started, take a look at the following example of a basic 4 x 4 table:

```
<table print-width="100%" screen-width="600px"
  align="left" id="tbl:example_4x4">
  <caption>
    Example of a simple 4x4 table without frame borders
  </caption>
  <shortcaption>
    Example of a 4x4 table (continued)
  </shortcaption>
  <colgroup>
    <col width="45%"/>
    <col width="55%"/>
  </colgroup>
  <tr>
    <td>First column, first row </td>
    <td>Second column, first row </td>
  </tr>
  <tr>
    <td>First column, second row </td>
    <td>Second column, second row</td>
  </tr>
</table>
```

The attributes `print-width` and `screen-width` determine the horizontal expansion of the table. For printable output, you can specify the table width in centimeters (cm), points (pt) or as a percentage (%) of the overall width of the page's text area. For HTML output, you can use all units commonly used in HTML Cascading Stylesheets or the statement `auto`, to leave the calculation of the table's dimensions completely to the browser. The function of the `align` attribute should be self-explanatory, it can take the values `left`, `center` and `right`. The `id` attribute gives the table a unique id, which can be referenced with the `ref` and `pageref` elements (see section 4.6).

The optional `caption` element can be used to give the table a descriptive annotation. If you supply a `shortcaption` it will be printed on continuing pages when a table extends over more than one page. The `colgroup` element describes the column layout. For each column in the table, there must be a `col` element specifying the relative width of the column. Make sure that these total up to 100%, or you may experience strange effects!

The table may start with one or more header rows distinguished by the `th` element and end with one or more footer rows distinguished by the `tfoot` element. Regular rows are set with the `tr` element, individual table cells with `td`. Table head and foot will be repeated on each page, if the table extends across multiple pages. Apart from that, no special formatting will be applied to text in header or footer cells.

5.2.2. Activating the Grid Rules

The table above does not have a visible grid. To draw a frame around your table, use the `frame` attribute on the table element and add an arbitrary combination of the keywords `left`, `right`, `top` and `bottom` to it, in a comma separated list. Each of the keywords turns on drawing of the respective line on the table's outer frame border.

Using the keywords `rowsep` and `colsep`, you can activate the dividing lines in between table cells. You can do this globally, by adding them to the table's `frame` attribute, or for individual rows and cells. Copy the following listing into an empty document and try adding and removing lines from the table grid, to get a feel for it.

```
<table print-width="100%" screen-width="600px"
  align="left" id="tbl:example_grid" frame="top,bottom"
  print-rulewidth="1pt" screen-rulewidth="1px" rulecolor="#000000">
  <colgroup>
    <col width="25%"/>
    <col width="25%"/>
    <col width="25%"/>
    <col width="25%"/>
  </colgroup>
  <th frame="rowsep">
    <td colspan="4"><b>Header</b></td>
  </th>
  <tr frame="colsep">
    <td frame="rowsep">1</td><td>2</td><td>3</td><td>4</td>
  </tr>
  <tr frame="colsep">
    <td>5</td><td frame="rowsep">6</td><td>7</td><td>8</td>
  </tr>
  <tr frame="rowsep,colsep">
    <td>9</td><td>10</td><td>11</td><td frame="rowsep">12</td>
  </tr>
  <tr>
    <td>13</td><td>14</td><td>15</td><td>16</td>
  </tr>
</table>
```

The thickness of the grid rules may be specified with the attributes `print-rulewidth` and `screen-rulewidth`. The color of the lines can be controlled via the `rulecolor` attribute. Color values must be given as CSS-style RGB triplets in hexadecimal notation. So in this example, the table rules would be black, which is also the default.

5.2.3. Coloring Table Cells

You may color individual rows or cells by setting the `color` attribute on the corresponding tag. For example, to give the second cell in the first row from the previous example a gray background, you could write:

```
<tr frame="colsep">
  <td frame="rowsep">1</td><td color="#dddddd">2</td><td>3</td><td>4</td>
</tr>
```

Please note that colored cells may appear to overlap with dark grid rules when viewing PostScript or PDF documents on screen. Therefore, you should avoid using colored cells and grid rules together or instead use white rules when working with colored tables.

5.2.4. Text-Alignment in Table Cells

The vertical alignment of text in tables can be controlled only for entire rows, but not for individual cells. This is due to \LaTeX 's limited capabilities in this respect. To determine the vertical alignment of text in a table row, set the `valign` attribute on the corresponding row element to one of the specifiers `top`, `middle` or `bottom`.

Horizontal text alignment can be controlled per row or for each cell individually, by setting the `align` attribute to `left`, `center` or `right`. Starting with ecromedos 2.0, text in tables can also be set justified. Per default, text is set left-aligned.

5.2.5. Rows Spanning Multiple Columns

Sometimes it may be necessary to make a table cell stretch across multiple columns. You can achieve this by setting the `colspan` attribute on a cell to the number of columns that it should cover. Unfortunately, there is no corresponding `rowspan` attribute, as it exists in HTML. However, in most cases it should be possible to work around this limitation using subtables.

5.2.6. Subtables

With ECML it is not possible to create tables with cells that span multiple rows, i.e. there is no `rowspan` attribute. Starting with version 2.0, you can use subtables to partially get around this limitation. A subtable is created by simply putting a `subtable` element in place of a `td` element. Here is an example:

```
<table print-width="100%" screen-width="600px"
  align="left" id="tbl:example_subtable" frame="top,left,right,bottom"
  print-rulewidth="1pt" screen-rulewidth="1px" rulecolor="#000000">
  <colgroup>
    <col width="25%"/>
    <col width="75%"/>
  </colgroup>
  <tr valign="middle">
```

```

<td align="center" frame="colsep">January 2009</td>
<subtable frame="colsep,rowsep">
  <colgroup>
    <col width="14%"/><col width="14%"/><col width="14%"/>
    <col width="14%"/><col width="14%"/><col width="15%"/>
    <col width="15%"/>
  </colgroup>
  <tr align="right">
    <td><b>Mon</b></td><td><b>Tue</b></td><td><b>Wed</b></td>
    <td><b>Thu</b></td><td><b>Fri</b></td><td><b>Sat</b></td>
    <td><b>Sun</b></td>
  </tr>
  <tr align="right">
    <td> </td><td> </td><td> </td><td> 1 </td>
    <td> 2 </td><td> 3 </td><td> 4 </td>
  </tr>
  <tr align="right">
    <td> 5 </td><td> 6 </td><td> 7 </td><td> 8 </td>
    <td> 9 </td><td> 10 </td><td> 11 </td>
  </tr>
  <tr align="right">
    <td> 12 </td><td> 13 </td><td> 14 </td><td> 15 </td>
    <td> 16 </td><td> 17 </td><td> 18 </td>
  </tr>
  <tr align="right">
    <td> 19 </td><td> 20 </td><td> 21 </td><td> 22 </td>
    <td> 23 </td><td> 24 </td><td> 25 </td>
  </tr>
  <tr align="right">
    <td> 26 </td><td> 27 </td><td> 28 </td><td> 29 </td>
    <td> 30 </td><td> 32 </td><td> </td>
  </tr>
</subtable>
</tr>
</table>

```

As you can see, the syntax for subtables is exactly the same as for regular tables, except that a subtable does not have an id or a caption and you cannot specify the table width, as it is fixed at 100%, stretching over the entire cell.

5.3. Embedding Graphics

Graphical figures are incorporated into a document via the `figure` element. You can give figures a caption and an id. A figure that carries an id attribute can be referenced via the `ref` and `pageref` elements (see Section 4.6). Take a look at the following example:

```

<figure align="center" id="fig:thebeach">
  <caption>The Beach</caption>

```

```

</figure>
<p>
  Figure <ref idref="fig:thebeach"/> shows a beautiful sunset at
  the Galveston Beach.
</p>
```

With the `src` attribute, you specify the location of the image on your harddisk. If the image's file format is not suitable for use with a particular output format, the document pre-processor will automatically convert it. For instance, when generating \LaTeX output, raster images are automatically converted to encapsulated postscript.

Important: *Supply images in a high-enough resolution for proper representation in all target formats.*

The attributes `print-width` and `screen-width` determine the width of the image in printable output and in XHTML output, respectively. For printable output this can be a value in points (pt) or centimeters (cm) or a percentage (%) of the overall width of the page's text area. The width for HTML output is specified in pixels (px).

The figure's horizontal alignment is controlled by setting the `align` attribute to `left`, `center` or `right`. If you would like a thin black border around your figure, set the `border` attribute to `yes`.

There is experimental support for letting the text flow around figures. Simply place the `figure` element *inside* a paragraph like an inline element and make sure that you explicitly set the figure's alignment to `left` or `right`.

You may also load small images or icons into the running text using the `img` element as an inline element.

5.4. About Floating Objects

Per default, figures and tables are placed exactly where specified in the source document. Imagine though, that you are generating printable output and so far the page has been filled by two thirds with text. Technically, the next thing to be inserted would be a picture, but it occupies more space than remains and thus has to be moved to the next page, leaving the page before empty by one third.

This is not only visually unpleasant, but also bloats your document unnecessarily. As a solution, you can turn figures or tables into *floating objects* by setting the `float` attribute on the main element to `yes`. Making an object float means that you give the formatting engine (i.e. \LaTeX) permission to move it to a different location in the text in order to warrant optimal text flow across pages.

5.5. Verbatim Text and Code Listings

You can use the `verbatim` element when you need to print scripts and want whitespace to be preserved. Text inside a verbatim tag will be printed in typewriter letters and whitespace will be displayed just as it appeared in your editor.

For program code you should use the `listing` element, which has as a single child the `code` element. You can have your code syntax highlighted by specifying the name of the programming language or script in the `syntax` attribute. Here is an example for the classic "Hello World" in C:

```
<listing>
  <code syntax="c" colorscheme="borland"
    strip="yes" startline="1" linestep="100"
    tabspaces="2"><![CDATA[
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    printf("Hello World!\n");
    return 0;
}
    ]]></code>
</listing>
```

ecromedos internally uses the powerful `Pygments` syntax highlighter, which can lex a vast amount of programming languages and scripts. `Pygments` also comes with a number of predefined coloring schemes (styles) that you may select with the `colorscheme` attribute. For a complete list of supported languages and available styles, run the command

```
#> pygmentize -L
```

If you specify a `startline`, the syntax highlighter will number each line in your code. The `linestep` attribute specifies the increment from one line to the next.

Setting the `strip` attribute on the `verbatim` or `code` elements to `yes` will result in whitespace being stripped from the beginning and end of your listing. You can override the background color of the selected coloring scheme with the `bgcolor` attribute.

Per default, ecromedos converts all tabulators inside a `verbatim` or `code` element to 4 spaces. You can override the number of spaces using the `tabspaces` attribute.

5.6. Mathematic Formulas

Mathematic formulas are entered in $\text{T}_\text{E}\text{X}$ notation. Explaining $\text{T}_\text{E}\text{X}$ is beyond the scope of this document. For more information, please refer to appropriate literature, such as [LSHORT].

5.6.1. Inline Math

In order to set mathematic expressions inline, i.e. in the running paragraph, use the `m` element. Take a look at this example:

```
<p>
  The main proposition of Einstein's
```

```
Theory of Relativity is that <m>e = mc^2</m>.  
</p>
```

5.6.2. Formulas as Block Elements

Formulas can also be set as block elements. Simply enclose the `m` element in an `equation` element. To have your equation numbered, set the `number` attribute to `yes`. The following listing shows, how to set the equation from the previous example as a block element:

```
<equation number="yes">  
  <m>e = mc^2</m>  
</equation>
```

Support for math is not yet very sophisticated and the presented mechanisms will not allow for more than the occasional mathematical one-liner. Future versions of ecromedos will provide better control over the alignment and grouping of formulas.

6. The Backmatter

This chapter deals with creating the backmatter of a document. The backmatter comprises the appendices, glossary and bibliography, as well as the keyword indexes.

6.1. The Appendices

An **appendix** is technically identical to a **chapter**, which also implies that you cannot have an appendix in an **article**. The only difference is that appendices are enumerated with latin letters. The appendices follow directly after the last chapter in a document. You can suppress an entry in the table of contents by setting the `tocentry` attribute to `no`.

6.2. Creating a Glossary

Creating a glossary can be as easy as pulling out your magic wand and saying `make-glossary`. If you are afraid, that spell might fire backwards, you can also create your glossary manually instead.

6.2.1. Making Definitions on the Spot

Whenever you come across a term or expression in your manuscript that you want to add to the glossary, use the `defterm` element to make the definition right there on the spot. The `defterm` element is a definition list (see also section 5.1.2) with only a single entry and it is used inline in the running paragraph as shown in this example:

```
<p>
Users who are already familiar with HTML or document markup
languages like DocBook

<defterm>
  <dt>DocBook</dt>
  <dd>
    DocBook is a semantic markup language, originally developed by HAL Computer
    Systems and O'Reilly & Associates to enable document interchange between
    document publications systems of different vendors. DocBook has since grown
    into a comprehensive document authoring system. Modern versions of DocBook
    use XML for the markup and come with templates based on the Extensible
    Stylesheet Language for the transformation of documents to representational
    formats.
  </dd>
</defterm>
```

```
will find learning ECML particularly easy.
</p>
```

There may be special cases, in which the collation rules of your locale have to be modified. This can be done indirectly by providing explicit sortkeys for glossary entries, using the `defterm` element's `sortkey` attribute. For example, in order to make “crocodile” sort as “krokodil”, you would write:

```
<defterm sortkey="krokodil">
  <dt>crocodile</dt>
  <dd>
    ...
```

6.2.2. Automatic Generation of the Glossary

Putting instances of `defterm` in your document will not automatically trigger the generation of a glossary section. For this to happen, you have to insert a `make-glossary` element right after the appendices, or after the final chapter if the document does not have an appendix. A typical occurrence of this element in a document written in American English might look like this:

```
...
</appendix>

<make-glossary locale="en_US" tocentry="yes" alphabet="[Symbols]
  A B C D E F G H I J K L M N O P Q R S T U V W X Y Z"/>

<biblio number="no">
...

```

The `alphabet` attribute holds a list of letters according to which your glossary will be *sectioned*. This has *nothing* to do with how the glossary will be *sorted*. In order to determine the collation, use the `locale` attribute instead. The `alphabet` attribute should contain a list of characters separated with whitespace. It may also contain a denominator of the form `[name]` – where *name* would typically be the word “symbols” – for entries that consist of only symbolic characters. The default alphabet is the basic latin alphabet.

The `locale` attribute indicates the language whose collation rules will be used to *sort* the glossary. Use one of the locale identifiers listed in section 3.3.1. Even though this attribute is optional, it is highly recommended that you always set it, or your glossary will be sorted according to the default locale of the system or account in which ecromedos is being run. If you select a locale, which is not supported on your system, ecromedos will fall back to standard C sorting.

The `tocentry` attribute, which can occur on any sectioning element, may be set to `no` to prevent the glossary from showing up in the table of contents (TOC), when generating printable output. In XHTML output, the glossary will *always* be listed in the TOC.

6.2.3. Creating the Glossary Manually

If you don't like the idea of having `defterm` elements all over your document, or for some other reason would prefer to set the glossary yourself, then here is how to do it:

```
<glossary>
...
  <glsection name="D">
    <dl>
      <dt>DocBook</dt>
      <dd>
        DocBook is a semantic markup language, originally developed by ...
      </dd>

      <dt>Doxygen</dt>
      <dd>
        A tool for documenting C++ code.
      </dd>
    </dl>
  </glsection>

  <glsection name="E">
    ...
  </glsection>
...
</glossary>
```

As you can see, the `glossary` is split into `glsections`. Typically, there will be a separate section for each letter of your alphabet, and the `name` of a section would be the letter that it represents. Each section contains a single definition list with all the entries for the respective letter. Simply put your glossary in place of the `make-glossary` element.

6.3. Keeping a Bibliography

Bibliographies are entered with the `biblio` tag and individual entries with `bibitem`. The bibliography comes after the last section in your document or after the glossary, if one exists. `ecromedos` does not support bibliographies for individual sections. A typical bibliography might look like this:

```
<biblio number="yes">
  <bibitem label="KOCH09" id="bib:KOCH09">
    Tobias Koch. <i>ecromedos 2.0 - Users' Manual</i>.
    <tt>ecromedos.net</tt>, 2009.
  </bibitem>
  <bibitem label="WALSH03" id="bib:WALSH03">
    Norman Walsh, Leonard Muellner.
    <i>DocBook: The Definitive Guide</i>.
    O'Reilly, 2003.
  </bibitem>
```

```
</biblio>
```

The `biblio` element encloses a series of `bibitem` elements, which represent the individual entries in the bibliography. Each `bibitem` carries a `label` and an `id` attribute. If the `number` attribute on the `biblio` element is set to `no`, then the user-supplied label will be shown next to the entry and in citations. If numbering is set to `yes`, which is the default, then the ordinal number of an item will be used as its label and the user-supplied identifier will be discarded.

Citing from the Bibliography

Use the `cite` element, when citing from a source listed in the bibliography, to point the reader to the corresponding entry like this:

```
...
<p>
For more information on DocBook, please refer to <cite idref="bib:WALSH03"/>.
</p>
...
```

The document pre-processor will replace each occurrence of the `cite` element with the label of the corresponding entry set in square brackets. For example, to refer to the first entry in the listing above, you would write `<cite idref="bib:KOCH09"/>`, which the document pre-processor would replace with `[1]` when numbering is turned on and with `[KOCH09]` when numbering is off.

6.4. Generating Keyword Indexes

ecromedos facilitates the generation of keyword indexes by allowing you to place index markers in the text as you are writing. These markers are collected, sorted and turned into corresponding index sections by the document pre-processor without requiring any additional effort from the writer. Just as for the glossary (see section 6.2.3), you can also set indexes manually, if required.

6.4.1. Placing Index Markers

Index markers are embedded inline into the running paragraph. However, in order to not cause too much visual disturbance in the source document, it is recommended to aggregate markers at the beginning or end of paragraphs. The listing below is an excerpt from the source code of this chapter:

```
<p>
  <idxterm group="ecml">
    <item>biblio</item>
    <subitem>number</subitem>
  </idxterm>
  <idxterm group="ecml">
    <item>bibitem</item>
  </idxterm>
  <idxterm group="ecml">
```

```

    <item>bibitem</item>
      <subitem>label</subitem>
</idxterm>
<idxterm group="ecml">
  <item>bibitem</item>
    <subitem>id</subitem>
</idxterm>

```

The `<tt>biblio</tt>` element encloses a series of `<tt>bibitem</tt>` elements, which represent the individual entries in the bibliography. Each `<tt>bibitem</tt>` carries a `<tt>label</tt>` and an `<tt>id</tt>` attribute. If the `<tt>number</tt>`

```

...
</p>

```

As you can see, you can arrange keywords into a hierarchy. In the example above, a marker for `bibitem` is placed on the base level, using the `idxterm` and `item` elements. Then the markers for `label` and `id` are added as `subitems`. There is also a `subsubitem` element, in case you ever need a third hierarchy level.

There is one thing to note about this example: all entries are placed in a separate `group` called `ecml`. Effectively, a separate index listing all ECML language elements is created. Entries for which no group is specified will go into the `default` group.

Analogously to the `defterm` element (see section 6.2.1), you can set an explicit `sortkey` on the `idxterm` element, if you need to tweak the collation in a special way.

6.4.2. Automatic Index Generation

Use the `make-index` element to have the index markers in a particular group assembled into an actual keyword index. This document, for example, contains an index that links all ECML language elements and attributes to the sections in which they are discussed. The generation of this index is triggered with a `make-index` element like this:

```

...
  </biblio>

  <make-index group="ecml" title="Language Elements" locale="en_US"/>
</report>

```

The `make-index` element goes logically after the `biblio` element.

The `locale` attribute indicates the language whose collation rules will be used to *sort* the index. Use one of the locale identifiers listed in section 3.3.1. Even though this attribute is optional, it is highly recommended that you always set it, or your index will be sorted according to the default locale of the system or account in which `ecromedos` is being run. If you select a locale, which is not supported on your system, `ecromedos` will fall back to standard C sorting.

The optional `title` attribute allows you to override the automatic title.

6.4.3. Manual Index Creation

Creating a full index single-handedly, can be a lot of work, but if you really must, here is how to do it: Instead of placing index markers in the document, you leave ordinary `labels` in places that you wish to reference from the index. Then in the index, simply use the `idxref` element, which behaves just like `pageref` (see section 4.6), to obtain the page number of a label. The following is an excerpt from the XML that the document pre-processor may have generated for the listing of ECML language elements that can be found at the end of this document.

```
<index title="Language Elements" columns="2" tocentry="yes">
  <idxsection name="A">
    <item>abstract <idxref idref="idx:label1"/></item>
    <item>appendix <idxref idref="idx:label2"/></item>
    <item>article <idxref idref="idx:label3"/></item>
    <subitem>bcor <idxref idref="idx:label4"/></subitem>
    <subitem>div <idxref idref="idx:label5"/></subitem>
    <subitem>lang <idxref idref="idx:label6"/></subitem>
    <subitem>papersize <idxref idref="idx:label7"/></subitem>
    <subitem>secnumdepth <idxref idref="idx:label8"/></subitem>
    <subitem>secsplitdepth <idxref idref="idx:label9"/>,
      <idxref idref="idx:label10"/></subitem>
  </idxsection>
  <idxsection name="B">
    ...
  </idxsection>
  ...
</index>
```

As you can see the structure of an index is somewhat similar to that of a glossary. The `index` is split into `idxsections`, each of which typically represents one letter of your alphabet. A section contains a series of `items`. Please note that `subitems` are *not* nested inside items. A `subitem` is considered as belonging to the `item` element immediately preceding it.

7. Styling Your Documents

For each supported output format, there exists a subfolder `transform/<format>` below the installation folder. Inside this folder, the XSL stylesheets with the transformation rules for this target format and a style definition file `style.xml` are stored.

To modify the style definitions for a given target format, create and edit a copy of the corresponding `style.xml` file. Then use the `-s` command line switch to point the document processor to the new style file. The format and contents of the style definition file differ depending on the output format.

7.1. Editing the CSS-Stylesheet

The central style definition for XHTML is an old-fashioned CSS stylesheet, wrapped inside an XML container. For users who have previous experience with CSS, most parts of the stylesheet should be straight-forward to figure out. The following subsections point out the elements of the stylesheet concerning the page layout.

7.1.1. Frontpage and Table of Contents

The document head, carrying for example the title, authors, date of publication and publisher is set inside a `div` section with id `#head`,

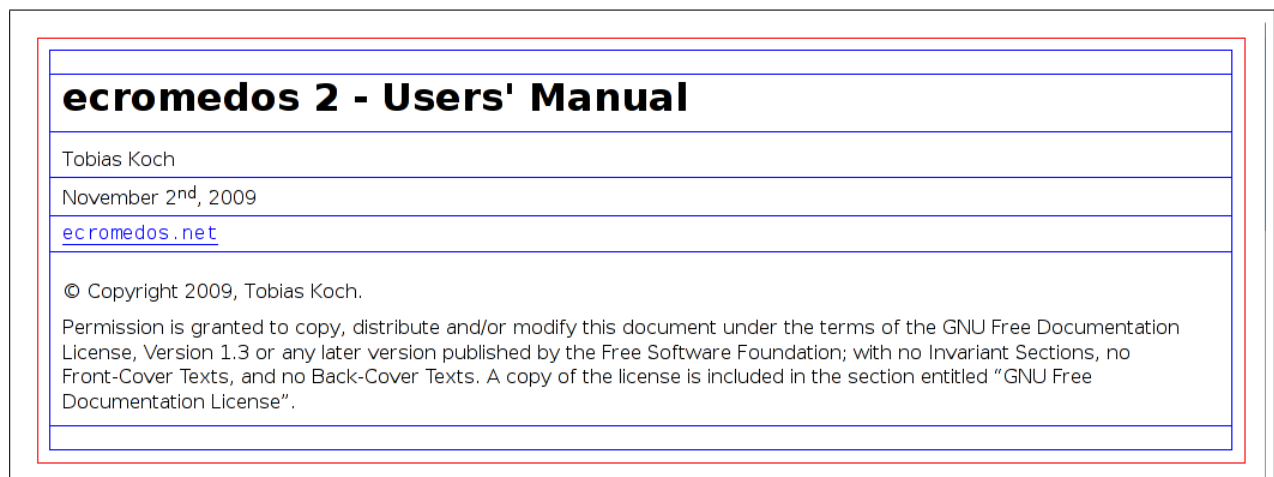


Figure 7.1: The title page header elements.

marked with a red outline in figure 7.1. The header elements themselves are arranged in a `table` with style class `head`, marked by the blue outline. Each element is contained in a separate cell with style class

head-element, e.g. **head-title**. The subtitle is set in the same cell as the main title, but wrapped in an extra **div** with style class **subtitle**, so that it can be styled differently from the main title.

Figure 7.2: The table of contents.

The table of contents (TOC) is contained inside a **div** with id **#toc**, marked with a red outline in figure 7.2. The TOC heading and “list of” headings are set in **divs** with style classes **toc-heading** and **listof-heading**, respectively. The individual entries in the TOC are **divs** with style classes **toc-mainitem** or **toc-subitem**, the empty elements in between sections have style class **toc-spacer**. The indentation of the entries cannot be influenced through the CSS stylesheet.

7.1.2. Text Body and Margin

The page contents are wrapped inside a **div** with id **#content**, marked with a red outline in figure 7.3. The main canvas and margin are table cells of classes **textbody** and **margin**, part of a table with style class **content**, marked with a blue outline.

Figure 7.3: The main canvas and margin.

7.1.3. Navigation Elements

The top navigation bar is set inside a **div** with id **nav-top**, which in turns contains a **table** with the same style class name. The table has a cell **nav-top-title** displaying the document title and a cell

`nav-top-buttons` containing the navigations buttons.

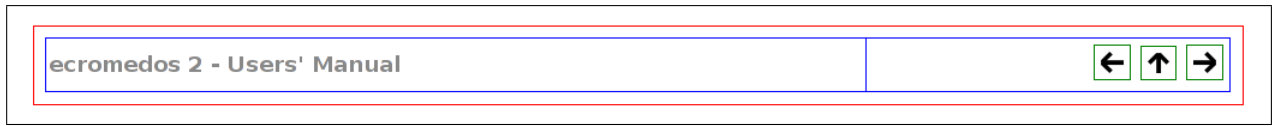


Figure 7.4: The top navigation bar.

Analogously, the bottom navigation bar consists of a `div` with id `nav-bottom` containing a table carrying the same style class name. This table is split into three cells `nav-bottom-left`, `nav-bottom-center` and `nav-bottom-right`. The text labels and links to previous, next and superordinate sections (see fig. 7.5) are styled in separate spans with style classes `nav-link-dir` (label) and `nav-link` (text).

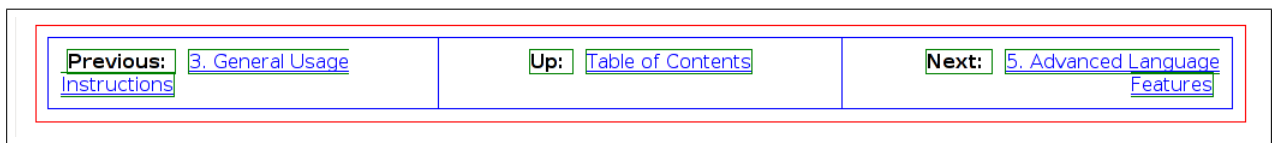


Figure 7.5: The navigation bar at the page bottom.

7.1.4. Glossary and Index

The heading at the top of the glossary page is a simple `h1` element. The heading at the beginning of each glossary section is also an `h1` with additional style class `glsection`. The alphabetic quick links at the top, which allow you to jump directly to certain sections are contained inside a `div` with class `gllinks`. Active links are spans of class `glink`, and inactive of class `glnolink`. The horizontal rule between sections has style class `glseparator`.

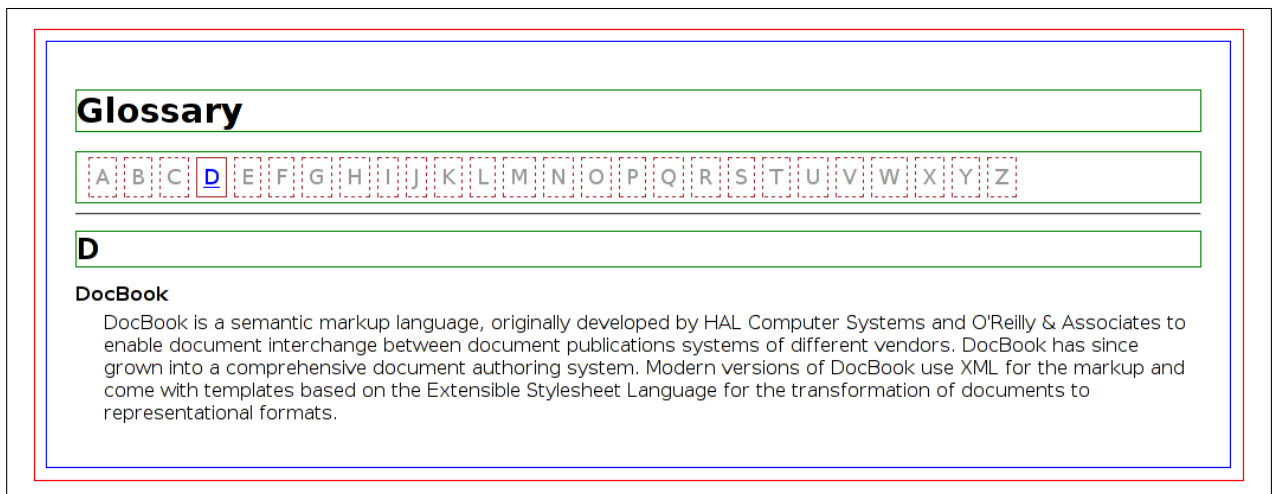


Figure 7.6: The glossary.

Analogously to the glossary, the heading atop an index is an `h1`, and the heading at the beginning of each section is an `h1` with additional style class `idxsection`. The alphabetic quick links are set in a

container with class `idxlinks` and are themselves spans of style classes `idxlink` and `idxnolink`. The entries are arranged in a table of class `idxlisting` whose cells are styled with class `idxitem`. Each text entry is wrapped inside a span of class `idxitem`, `idxsubitem` or `idxsubsubitem`; indentation is controlled through these elements.

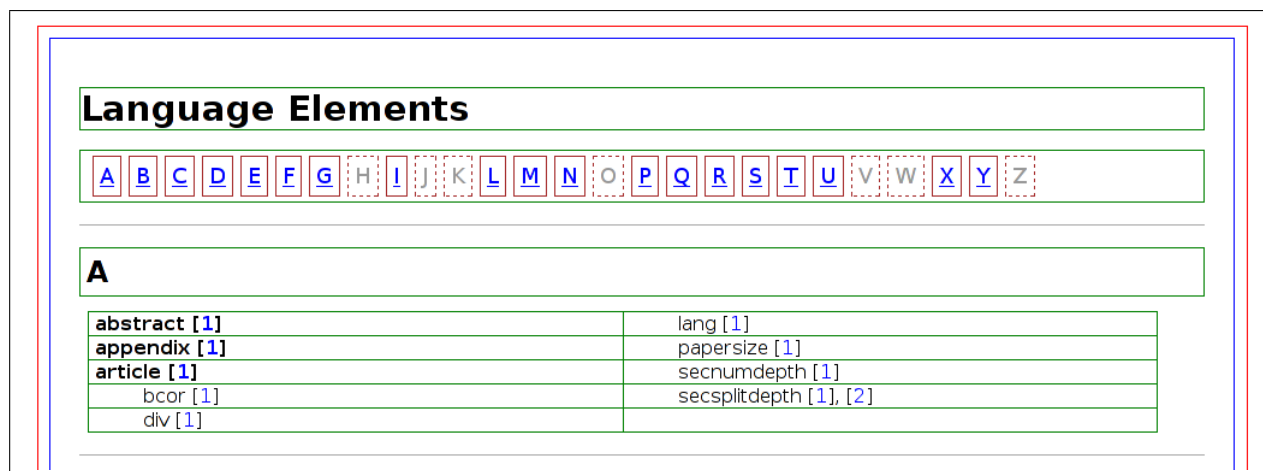


Figure 7.7: A keyword index.

7.2. Changing the Look of L^AT_EX-based Output

In the style definition file for L^AT_EX, you will find a separate section for each of the supported document classes, with each section containing a list of stylable elements for the corresponding document class. Some basic knowledge of L^AT_EX will be required to be able to edit these.

The following style elements exist:

caption-setup

The package options that are passed through to the caption L^AT_EX package. Please consult [CAPTION], pages 5 ff., for more information on the available options.

chapterpage-style

The page style to be used for pages starting a new chapter (see [KOMASCR], pages 66 ff.). Use `empty` for a page without any decorations, `scrplain` for a page that is undecorated but has a page number, or `scrheadings` for a page with page number and column titles.

chapter-title

The font properties of a chapter title.

description-label

The font properties of a label marking up a term in a definition list.

document-font

Use this element to change the font settings. For instance, in order to set your document in a sans-serif font family, you could set this element to

```
\renewcommand{\rmdefault}{\sfdefault}\normalfont
```

document-options

The package options that are passed to the \LaTeX macro package KOMA-Script. Please consult [KOMASCR], pages 43 ff., for more information on the available options.

document-subtitle

The font properties of the document subtitle.

document-title

The font properties of the document title.

footnote

The font properties of a footnote.

footnote-label

The font properties of a label in a footnote.

footnote-reference

The font properties of a footnote reference.

indexpage-style

The page style to be used for pages starting a keyword index (see [KOMASCR], pages 66 ff.). Use `empty` for a page without any decorations, `scrplain` for a page that is undecorated but has a page number, or `scrheadings` for a page with page number and column titles.

minitoc-chapter-title

Font properties of the title of a chapter overview.

minitoc-part-title

Font properties of the title of a part overview.

minitoc-section-title

Font properties of the title of a section overview.

page-head

Font properties of the column titles of a page of style `scrheadings` (see [KOMASCR], page 62).

page-head-fields

This element is meant to contain the definitions of the column titles for pages of style `scrplain` and `scrheadings`. For a detailed description of the contained elements, consult [KOMASCR], pages 127 ff.

page-number

Font properties for page numbers.

page-style

Sets the default page style to one of `empty`, `scrplain`, `scrheadings` (see [KOMASCR], pages 126 ff.).

paragraph-title

Font properties of an inline paragraph title.

part-number

Font properties of a part number.

partpage-style

The page style to be used for pages starting a new part (see [KOMASCR], pages 66 ff.). Use `empty` for a page without any decorations, `scrplain` for a page that is undecorated but has a page number, or `scrheadings` for a page with page number and column titles.

part-title

Font properties of a part title.

sectioning-title

Default font properties for section titles. This affects, for example, the `minisection` titles, for which no separate style element exists.

section-title

Font properties of the title of a `section` element.

subsection-title

Font properties of the title of a `subsection` element.

subsubsection-title

Font properties of the title of a `subsubsection` element.

titlepage-style

The page style to be used for the title page (see [KOMASCR], pages 66 ff.). Use `empty` for a page without any decorations, `scrplain` for a page that is undecorated but has a page number, or `scrheadings` for a page with page number and column titles.

A. GNU General Public License

Version 2, June 1991

Copyright © 1989, 1991 Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software - to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps:

1. copyright the software, and
2. offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program

proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: If the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them

as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any

work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>

Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the

Free Software Foundation, Inc.

**51 Franklin Street, Fifth Floor
Boston, MA 02110-1301, USA**

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type "show w". This is free software, and you are welcome to redistribute it under certain conditions; type "show c" for details.

The hypothetical commands "show w" and "show c" should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than "show w" and "show c"; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program "Gnomovision" (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

B. GNU Free Documentation License

Version 1.3, 3 November 2008

Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc. fsf.org

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain

any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, \LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “publisher” means any person or entity that distributes copies of the Document to the public.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section

of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

“Massive Multiauthor Collaboration Site” (or “MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A “Massive Multiauthor Collaboration” (or “MMC”) contained in the site means any set of copyrightable works thus published on the MMC site.

“CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

“Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is “eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (C) YEAR YOUR NAME.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled “GNU Free Documentation License”.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with ... Texts.” line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Bibliography

- [W3CXML] World Wide Web Consortium. *Extensible Markup Language (XML) 1.0 (Third Edition)*. W3C Recommendation. February 4, 2004.
URL: <http://www.w3.org/TR/2004/REC-xml-20040204/>.
- [W3CXSL] World Wide Web Consortium. *Extensible Stylesheet Language (XSL) Version 1.0*. W3C Recommendation. October 15, 2001.
URL: <http://www.w3c.org/TR/xsl/>.
- [W3CXSLT] World Wide Web Consortium. *XSL Transformations (XSLT) Version 1.0*. W3C Recommendation. November 16, 1999.
URL: <http://www.w3c.org/TR/xslt>.
- [DBGUIDE] Norman Walsh. *DocBook 5.0: The Definitive Guide*. Draft Version 0.0.25, Mar 2008.
URL: <http://www.docbook.org/tdg5/en/html/docbook.html>
- [LSHORT] Tobias Oetiker et al. *The Not so Short Introduction to L^AT_EX 2_ε*. Version 4.20, May 2006.
URL: <http://www.ctan.org/get/info/lshort/english/lshort.pdf>
- [LATEX] Wikipedia The Free Encyclopedia. *L^AT_EX*.
URL: <http://en.wikipedia.org/wiki/LaTeX>.
- [GPL2] The Free Software Foundation. *The GNU General Public License, version 2*. June 1991.
URL: <http://www.gnu.org/licenses/old-licenses/gpl-2.0.html>.
- [KOMASCR] Frank Neukam, Markus Kohm, Axel Kielhorn. *The KOMA-Script Bundle*. June 2009.
URL: <ftp://ftp.ctan.org/tex-archive/macros/latex/contrib/koma-script/scrguien.pdf>.
- [CAPTION] Axel Sommerfeldt. *Customizing captions of floating environments using the caption package*. March 2008.
URL: <http://tug.ctan.org/tex-archive/macros/latex/contrib/caption/caption-eng.pdf>

Language Elements

A

abstract	21
appendix	36
tocentry	36
article	13
bcor	17
div	17
lang	16
papersize	17
parskip	17
secnumdepth	17
secsplitdepth	16
author	21

B

b	22
bibitem	39
id	39
label	39
biblio	38
number	39
blockquote	25
book	13
bcor	17
div	17
lang	16
papersize	17
parskip	17
secnumdepth	17
secsplitdepth	16
br	23

C

caption	29, 32
chapter	20
cite	39
idref	39
code	33

bgcolor	34
colorscheme	34
linestep	34
startline	34
strip	34
syntax	33
tabspace	34
col	29
width	29
colgroup	29
color	22
rgb	22
counter	24

D

date	21
dd	28
defterm	36
sortkey	37
dl	28
dt	28

E

entity	25
name	26
equation	35
numbering	35

F

figure	32
align	33
float	33
id	32
footnote	25

G

glossary	21, 38
glsection	38

name 38

H

head 21

I

i 22

idxref 41

idxsection 41

 name 41

idxterm 39

 group 40

 sortkey 40

img 33

 print-width 33

 screen-width 33

 src 33

index 41

item 40

L

label 23

 id 23

large 22

legal 21

li 27

listing 33

m 34

M

make-glossary 37

 alphabet 37

 locale 37

 toceentry 37

make-index 40

 group 40

 locale 40

 title 40

make-toc 17

 depth 17

 lof 17

 lol 17

 lot 17

marginal 24

medium 22

minisection 21

N

nobr 23

O

ol 27

 type 27

P

p 21

pagebreak 23

pageref 24

 idref 24

part 17

preface 21

publisher 21

Q

q 25

qq 25

R

ref 24

 idref 24

report 13

 bcor 17

 div 17

 lang 16

 papersize 17

 parskip 17

 secnumdepth 17

 secsplitdepth 16

S

section 20

shortcaption 29

small 22

sub 22

subitem 40

subject 21

subsection 20

subsubitem 40

subsubsection 20

subtable 31

 color 31

 colspan 31

frame31
 subtitle 21
 sup 22

T

table 29
 align 29
 float 33
 frame 30
 id 29
 print-rulewidth 30
 print-width 29
 rulecolor 30
 screen-rulewidth 30
 screen-width 29
 td 29
 align 31
 color 31
 colspan 31
 frame 30
 tf 29
 align 31
 color 31
 frame 30
 valign 31
 th 29
 align 31
 color 31

frame30
 valign 31
 title 21
 tr 29
 align 31
 color 31
 frame30
 valign 31
 tt 22

U

u 22
 ul 27

V

verbatim 33
 strip 34
 tabspaces 34

X

x-large 22
 x-small 22
 xx-large 22
 xx-small 22

Y

y 22