ARAKHNE.ORG
Another Yet Free Software Gateway

LaTeX PACKAGES FOR STRUCTURED DOCUMENTS AS FOR UNIFIED PROCESS METHODOLOGY

LaTeX Packages for Structured Documents as for Unified Process Methodology

Official Documentation

Reference:	UPM-2025-01
Version:	34.0
Updated:	2025/04/29
Status:	Public

Authors: Stéphane Galland , Frans van Dunné

This document describes the LaTeX Packages for Structured Documents as for Unified Process Methodology project.

TeX and LaTeX are a trademarks of the American Mathematical Society.

`tex-upmethodology` is owned by Stéphane Galland, Arakhnê.org, France.

This document was realised with LaTeX and `tex-upmethodology`.

Reference : UPM-2025-01

Document Summary

Project	LaTeX Packages for Structured Documents as for Unified Process Methodology
Document	Official Documentation
Reference	UPM-2025-01
Version	34.0
Last Update	2025/04/29

Authors

Names	*Comments*	*Emails*
Stéphane Galland	Original Author	galland@arakhne.org
Frans van Dunné	Reviewer	

Validators

Names	*Comments*	*Emails*	*Initials*
Stéphane Galland	Original Author	galland@arakhne.org	

Version History

Version	*Date*	*Updates*
25.0	2025/03/14	Update the documentation for the `declareupmtheorem` command and `definition` environment.
25.1	2025/03/15	Add the mention of the dependency to `tobibind` in the dependencies of `upmmethodology-fmt`.
26.0	2025/03/15	Add the commands `\defref` and `\defpageref`.
27.0	2025/03/15	Add the optional ”source text” to the commands `\mfigure` and `\mfigure*`.
28.0	2025/03/15	Add the commands `\addsource` and `\tablenote` for the `mtable` environment.
29.0	2025/03/15	Use the package `tcolorbox` for implementing `mtabular` and `mtable` environments.
30.0	2025/03/23	Add documentation of the algorithms’ colors.
31.0	2025/04/06	Add macro `\upmcaption`.
32.0	2025/04/12	Replace `tabularx` by `xltabular` and table options `figurecaptionabove` and `figurecaptionabove`.
33.0	2025/04/20	Extend the internal value definition API.
33.1	2025/04/27	Add option `pgf` for figure commands.
33.2	2025/04/27	Include package `enumitem` for inline enumerations.
33.3	2025/04/27	Enable multiple lines the column heads with `\tabularheader`.
34.0	2025/04/29	Add environment `mlongtable`.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

1

INTRODUCTION

This documentation is written for and compiled by the
version 20250429
of tex-upmethodology.

This set of packages enables users to write documents according to the Unified Process Methodology. It was initially written by Stéphane GALLAND from the laboratory "Systèmes et Transports"[1] and is distributed by the Arakhnê.org website. The provided packages and classes may also be used for other types of documents (reports, theses...). Since 2012, it is used to support the layout and the style for the PhD theses of the Doctoral School SPIM[2].

Packages are:

- `upmethodology-version.sty`: makes it possible to set the version and the status of the document. It also makes it possible to manage the document history;
- `upmethodology-fmt.sty`: provides some useful functions to format the UP documents;
- `upmethodology-document.sty`: provides functions to manage the project, the subproject and the status of the document;
- `upmethodology-frontpage.sty`: formats and provides a front page for the document;
- `upmethodology-backpage.sty`: formats and provides a back page for the document;
- `upmethodology-task.sty`: is the *optional* LaTeX 2$_{\varepsilon}$ package that provides commands to manage project's tasks.
- `upmethodology-spec.sty`: is the *optional* LaTeX 2$_{\varepsilon}$ package that provides commands to build a specification description.
- `upmethodology-document.cls`: is the LaTeX 2$_{\varepsilon}$ class that provides the whole document specification. It is based on `book` and on the previous packages;
- `upmethodology-code.sty`: provides *optional* commands for source code formatting;
- `upmethodology-extension.sty`: provides commands for extension mechanism.

---

[1] Laboratory *Systèmes et Transport* (IRTES-SET), Institut de Recherche sur le Transport, l'Énergie et la Société (ITRTES), Université de Technologie de Belfort-Montbéliard (UTBM), France, http://set.utbm.fr/

[2] Doctoral School on the Sciences for engineers, and microtechnics, http://ed-spim.univ-fcomte.fr/

I

General User Documentation

2

Download and Installation

This chapter describes where to download `tex-upmethodology` and how to install it.

2.1 Download

`tex-upmethodology` is available on the *Arakhnê*.ORG website: http://www.arakhne.org/tex-upmethodology/.

Different types of installation are available: manual installation, Debian packages.

2.2 Manual System-wide Installation

To make `tex-upmethodology` available to all users, copy the content of the `tex-upmethodology` archive inside one of your system texmf directory, usually one of:

- `/usr/share/texmf-texlive/tex/latex/upmethodology`,
- `/usr/share/texmf/tex/latex/upmethodology`.

The second is to rebuild the LaTeX databases by invoking on a console (Unix syntax us used):

```
$> sudo mktexlsr
$> sudo update-updmap --quiet
```

`sudo` is a standard Linux tool that allows authorized users to temporarily obtain the administration rights.

2.3 Manual User-wide Installation

To make `tex-upmethodology` available to one user, copy the content of the `tex-upmethodology` archive inside the `$HOME/texmf` directory.

It is not required to rebuild the system-wide LaTeX databases because the user's texmf are dynamically parsed by the LaTeX distributions.

2.4 Debian Package Installation

Debian packages are available on Arakhnê.org website: http://www.arakhne.org/ubuntu.html. Please follow the given rules.

2.5 Package Dependencies

This section contains the list of all the package dependencies for the `upmethodology` packages.

2.5.1 upmethdology-backpage.sty

`upmethodology-backpage` package depends on:

- `upmethodology-extension`
- `upmethodology-p-common`

2.5.2 upmethdology-code.sty

`upmethodology-code` package depends on:

- `upmethodology-p-common`

2.5.3 upmethdology-document.cls

`upmethodology-document` class depends on:

- `a4wide`
- `fancyhdr`
- `hyperref`
- `upmethodology-backpage`
- `upmethodology-code` (optional)
- `upmethodology-document`
- `upmethodology-extension`
- `upmethodology-frontpage`
- `upmethodology-p-common`
- `upmethodology-spec` (optionnal)
- `upmethodology-task` (optionnal)
- `url`

2.5.4 upmethdology-document.sty

`upmethodology-document` package depends on:

- `babel`
- `draftwatermark` (only if the document is marked as ”draft” or ”restricted”)
- `upmethodology-extension`
- `upmethodology-fmt`
- `upmethodology-p-common`
- `upmethodology-version`
- `vmargin`

2.5.5 upmethdology-extension.sty

`upmethodology-extension` package depends on:

- `upmethodology-p-common`

2.5.6 upmethdology-fmt.sty

`upmethodology-fmt` package depends on:

- `bbm`
- `amsmath`
- `amsthm`
- `colortbl`
- `enumitem`
- `environ`
- `fontawesome5`
- `graphicx`
- `hyphenat`
- `longtable`
- `multicol`
- `picinpar`
- `pifont`
- `relsize`
- `setspace`
- `subcaption`
- `tabularx`
- `tcolorbox`
- `thmtools`

- `tikz`
- `titlesec`
- `tocbibind`
- `txfonts`
- `upmethodology-p-common`
- `varioref`
- `xcolor`
- `xltabular`
- `xkeyval`

2.5.7 upmethdology-frontpage.sty

`upmethodology-frontpage` package depends on:

- `upmethodology-document`
- `upmethodology-extension`
- `upmethodology-p-common`

2.5.8 upmethdology-p-common.sty

`upmethodology-p-common` package depends on:

- `ifpdf`
- `ifthen`
- `xcolor`
- `xspace`

2.5.9 upmethdology-spec.sty

`upmethodology-spec` package depends on:

- `ulem`
- `upmethodology-code`
- `upmethodology-fmt`
- `upmethodology-p-common`

2.5.10 upmethdology-task.sty

`upmethodology-task` package depends on:

- `upmethodology-p-common`
- `upmethodology-version`

2.5.11 upmethdology-version.sty

`upmethodology-version` package depends on:

- `upmethodology-fmt`
- `upmethodology-p-common`

II

PACKAGE DOCUMENTATION

3

CLASS UPMETHODOLOGY-DOCUMENT

Version: 2025/04/20

The LaTeX class `upmethodology-document` provides the basic configuration for a document. According to an option, this class is able to extend the standard `book`, `report` or `article` LaTeX classes. It also include several of the other `upmethdology` packages.

3.1 Types of documents

`upmethodology-document` supports three particular options, which permit to set the type of document:

- `book`: A book-specification is a two-sided document composed of parts and chapters, and with a copyright page and document information page. This option indicates to `upmethodology-document` to load the LaTeX standard `book` class. In addition the `\part` and `\chapter` commands are supported, and the following commands are automatically expanded: `\makefrontcover`, `\upmpublicationpage`, `\upmdocumentsummary`, `\makebackcover`. This behaviour may be overridden by the other class options.
- `report`: A report-specification is a one-sided document composed of chapters (no part), and with a document information page. This option indicates to `upmethodology-document` to load the LaTeX standard `report` class. In addition the `\part` command is ignored[1] and `\chapter` command is supported, and the following commands are automatically expanded: `\makefrontcover`, `\upmdocumentsummary`, `\makebackcover`. This behaviour may be overridden by the other class options.
- `article`: An article-specification is a one-sided document composed of sections (no part nor chapter). This option indicates to `upmethodology-document` to load the LaTeX standard `article` class. In addition the `\part` and `\chapter` commands are ignored[1], and the following commands are automatically expanded: `\makefrontcover`, `\makebackcover`. This behaviour may be overridden by the other class options.

---

[1] The command is redefined to print a warning message when used, no error message is generated.

3.2 Class options

The tables 3.1 and 3.2 contain the options supported by `upmethodology-document`. Any option not explicitely supported by the class is directly passed to the underlying standard LaTeX class (`book`, `report` or `article` according to the type of document, see 3.1).

3.3 Additional Features

`upmethodology-document` provides a constant behaviour for all types of document:

- `hyperref` is loaded and set with the document informations;
- `\setpdfcolor` is redefined and linked to `hyperref`;

Cat.	*Option*	*Explanation*
Document Type	article	see section 3.1
	book	see section 3.1.
	report	see section 3.1
Page Type	oneside	the document is generated assuming that each page will be printed on its recto side. This option overrides any previous occurrence of `twoside` option
	twoside	the document is generated assuming that each page will be printed on both recto and verso sides. This option overrides any previous occurrence of `oneside` option
Language	english	the document is written in English. `upmethodology` packages use the English translations for the generated texts. This option overrides any previous occurrence of `french` option
	francais	same as `french`
	french	the document is written in French. `upmethodology` packages use the French translations for the generated texts. This option overrides any previous occurrence of `english` option
Informations	documentinfo	invoke `\upmdocumentsummary`, `\upmdocumentauthors`, `\upmdocumentvalidators`, `\upmdocumentinformedpeople`, and `\upmhistory` commands at the begining of the document. This option overrides any previous occurrence of `nodocumentinfo` option
	nodocumentinfo	do not invoke `\upmdocumentsummary`, `\upmdocumentauthors`, `\upmdocumentvalidators`, `\upmdocumentinformedpeople`, nor `\upmhistory` commands at the begining of the document. This option overrides any previous occurrence of `documentinfo` option
	nopubpage	do not invoke `\upmpublicationpage` command at the begining of the document. This option overrides any previous occurrence of `pubpage` option
	pubpage	invoke `\upmpublicationpage` command at the beginning of the document. This option overrides any previous occurrence of `nopubpage` option

Table 3.1: Options (1/2) of `upmethodology-document` class

Cat.	*Option*	*Explanation*
Format	backcover	put the cover page at the end of the document
	figurecaptionabove	By default, the labels of the figures are below the figures. This option permits to put the figure labels above the figures
	frontcover	put the cover page at the beginning of the document
	frontmatter	invoke `\frontmatter` (and other related commands)
	nobackcover	do not put a cover page at the end of the document
	nofrontcover	do not put a cover page at the beginning of the document
	nofrontmatter	do not invoke `\frontmatter` (and other related commands)
	standardlist	disable the override of the lists (enumeration, etc.) for restoring the standard LaTeX lists
	standardlists	The style does not override the standard list, description and enumeration definitions
	tablecaptionbelow	By default, the labels of the tables are below the tables. This option permits to put the table labels above the tables
Packages	codepackage	Include the `upmethodology-code` package
	specpackage	Include the `upmethodology-spec` package
	taskpackage	Include the `upmethodology-task` package

Table 3.2: Options (2/2) of `upmethodology-document` class

4

PACKAGE UPMETHODOLOGY-VERSION

`Version: 2025/04/12`

The package `upmethodology-version` makes it possible to set the version and the status of the document. It also provides functions to manage the document history;

4.1 Constants for the Document Status

Some LaTeX 2_ε variables provides strings that describe the status of the document. They can be used in functions such as `\updateversion`.

- `\upmrestricted`: the document is under a restricted access, generally corresponding to the list of authors;
- `\upmvalidable`: authors indicates with this flag that the document could be sent to validators;
- `\upmvalidated`: the document was validated, but not published;
- `\upmpublic`: the document published and accessible to all people;

4.1.1 Information about the Document

The following functions permit to access to the informations about the document:

- `\theupmversion`: replies the last version number for the document;
- `\upmdate{version}`: replies the updating date of the document corresponding to the given version number;
- `\upmdescription{version}`: replies the updating comment of the document corresponding to the given version number;
- `\upmstatus{version}`: replies the status of the document corresponding to the given version number.
- `\theupmdate`: replies the last updating date for the document. It is equivalent to `\upmdate{\theupmversion}`;

- `\theupmlastmodif`: replies the last updating comment for the document. It is equivalent to `\upmdescription{\theupmversion}`;

- `\theupmstatus`: replies the last status for the document. It is equivalent to `\upmstatus{\theupmversion}`;

4.2 Register Revisions

The package `upmethodology-version` makes it possible to register revisions for building an history. The available functions are:

- `\updateversion{version}{date}{description}{status}`: registers a revision for the document. The revision indicates that the given version was produced at the given date. A small description of the changes and the resulting document's status must be also provided. The function `\updateversion` is a generalization of the following functions;

- `\initialversion[version]{date}{description}{status}`: registers the initial version of the document. If not given, the version is assumed to be `0.1`;

- `\incversion{date}{description}{status}`: registers a revision corresponding to the next major version. For example, if the version number was `2.67` before `\incversion`, this function add the version `3.67` with the given informations (incrementation of the major part of the version number);

- `\incsubversion{date}{description}{status}`: registers a revision corresponding to the next minor version. For example, if the version number was `2.67` before `\incsubversion`, this function add the version `2.68` with the given informations (incrementation of the minor part of the version number);

4.3 Formatted List of Versions

To obtain a formatted list of versions, you could use the command `\upmhistory[width]` which produces:

Version History

Version	*Date*	*Updates*
25.0	2025/03/14	Update the documentation for the `declareupmtheorem` command and `definition` environment.
25.1	2025/03/15	Add the mention of the dependency to `tobibind` in the dependencies of `upmmethodology-fmt`.
26.0	2025/03/15	Add the commands `\defref` and `\defpageref`.
27.0	2025/03/15	Add the optional ”source text” to the commands `\mfigure` and `\mfigure*`.
28.0	2025/03/15	Add the commands `\addsource` and `\tablenote` for the `mtable` environment.
29.0	2025/03/15	Use the package `tcolorbox` for implementing `mtabular` and `mtable` environments.
30.0	2025/03/23	Add documentation of the algorithms’ colors.
31.0	2025/04/06	Add macro `\upmcaption`.
32.0	2025/04/12	Replace `tabularx` by `xltabular` and table options `figurecaptionabove` and `figurecaptionabove`.
33.0	2025/04/20	Extend the internal value definition API.
33.1	2025/04/27	Add option `pgf` for figure commands.
33.2	2025/04/27	Include package `enumitem` for inline enumerations.
33.3	2025/04/27	Enable multiple lines the column heads with `\tabularheader`.
34.0	2025/04/29	Add environment `mlongtable`.

4.4 Localization

The following commands defines some localized strings used by `upmethodology-version`:

- `\upm@lang@date`: Date;
- `\upm@lang@updates`: Updates;
- `\upm@lang@version`: Version;
- `\upm@lang@version@history`: Version History;

5

PACKAGE UPMETHODOLOGY-FMT

`Version: 2025/04/29`

The package `upmethodology-fmt` provides some useful facilities to format a document.

5.1 Default Configuration for the Package `graphicx`

The package `graphicx` is included, and the following configuration is applied:

- **Image extensions:** By default, the supported image extensions are, in the preference order: `pdf`, `png`, `jpg`, `jpeg`, `tiff`, `gif`. Note that, the `tiff` picture format is not always supported by the TEX tools.

  To redefine these extensions, you must invoke:
  `\DeclareGraphicsExtensions{extensions}`
  where `extensions` must be replaced by a list of extensions separated by comas.
  Example: `\DeclareGraphicsExtensions{.pdf,.png,.eps}`

- **Image search path:** By default, the images are search inside the path "`./`". To redefine the search paths, you must invoke: `\graphicspath{{path1},{path2},{path3}...}`
  where `path1`, `path2`, `path2`, etc. must be replaced by the names of the directories in which the images are located. The paths in the list are separated by comas. *Do not forget to write a slash or a backslash character (depending on the path naming conventions for your operating system) at the end of each path.*
  Example: `\graphicspath{{./imgs/},{./imgs/auto/}}`

5.2 Contextual Search Path for `graphicx`

As described into the previous section, the `graphicx` package is able to search for files into a set of defined paths.

In order to define a search path that is valid for a part of the document, the `graphicspathcontext` environment is defined. This environment redefines the `graphicx` path with the environment's parameter. The original value of the `graphicx` path is restored when existing of the environment.

Syntax.

```
\begin{graphicspathcontext}{path}
...
\end{graphicspathcontext}
```

The parameter `path` must follow the syntactic definition of the `graphicx` path. If you want to reuse the current value of the `graphicx` path, you could obtain it by using the `\old` command.

Syntax with `\old`.

```
\begin{graphicspathcontext}{mypath,\old}
...
\end{graphicspathcontext}
```

Note that `\old` *must not be inside curly braces.*

5.3 Extended Caption for Floats with Explanation and Source

In LaTeX, a *float* refers to a specialized environment used to handle content such as figures, tables, and other elements that do not fit seamlessly within the regular flow of text. Floats are designed to ”float” to optimal positions in the document, ensuring proper formatting and avoiding awkward breaks in the text.

Floats also include captions and numbering for easy referencing within the text. By default, captions are placed below figures and above tables, but this can be customized. The captions of the floats also appear in the lists of floats, such as `\listoffigures` and `\listoftables`.

One could specify different texts for the captions on the side of the float itself and in these lists. The standard command `\caption` takes a mandatory argument that corresponds to the text on the side of the float, and an optional argument that corresponds to the text in the lists.

5.3.1 Explanation and Sources

Nevertheless, two types of usefull informations may be attached to a float:

- **Detailed Explanation:** it is a text that provides a detailed explanation on the float content. This text is not usefull when it appears in the lists of floats. It is usefull only one the side of the float itself.
- **Source:** Some floats refer to other documents: the sources. They corresponds to the original source of a figure or table, or a document that originally explained the content of the float. It is interesting to attach these sources to the floats, but not to see them in the lists of floats.

5.3.2 Extended Caption Command

To render these two types of information in a different way as the regular float's caption, the package provides the following command, which extends the LaTeX `\caption` command, and takes three (3) arguments:

Figure 5.1: Example of figure with explanation
This figure illustrates the rendering of the explanation attached to a figure

```
\upmcaption{⟨caption⟩}{⟨explanation⟩}{⟨source⟩}
```

Where:

- ⟨caption⟩ is the text of the regular caption that will appear on the side of the float and in the list of floats;
- ⟨explanation⟩ is the text that provides detailed explanation about the content of the float. This explanation will appear on the side of the float but not in the list of floats;
- ⟨source⟩ is the text that provides sources for the float. These sources will appear on the side of the float but not in the list of floats.

5.3.3 Example with Explanation Only

Let the following LaTeX code for including a figure with an explanation only (no source text):

```
\begin{figure}
\centering
\includegraphics{figure.pdf}
\upmcaption{Example of figure with explanation}{This figure illustrates the
rendering of the explanation attached to a figure}{}
\end{figure}
```

Figure 5.1 is the result of this example.

5.3.4 Example with Source Only

Let the following LaTeX code for including a figure with a source text only (no explanation):

```
\begin{figure}
\centering
\includegraphics{figure.pdf}
\upmcaption{Example of figure with source text}{}{This is the source of the figure}
\end{figure}
```

Figure 5.2 is the result of this example.

Figure 5.2: Example of figure with source text

Source:This is the source of the figure

5.3.5 Example with Explanation and Source

Let the following LaTeX code for including a figure with an explanation and a source text:

```
\begin{figure}
\centering
\includegraphics{figure.pdf}
\upmcaption{Example of figure with source text}{This figure illustrates the
rendering of the explanation and a source text that are attached to a figure}{This
is the source of the figure}
\end{figure}
```

Figure 5.3 is the result of this example.

5.4 Figures

It may be verbose to put LaTeX code to include a figure inside your document. To simplify your life, you could include a figure with the following commands.

Syntax 1.

```
\mfigure[position]{include_graphics_options}{filename}{caption}{label}[source text]
```

Syntax 2.

```
\mfigure*[position]{include_graphics_options}{filename}{caption}{label}[source text]
```

The difference between `\mfigure` and `\mfigure*` is the same as the difference between `\begin{figure}` and `\begin{figure*}`: the star-version fits to the entire paper width event if the document has two or more columns.

5.4.1 Commands for Including Figures

These two commands make it possible to include an image in your document. The parameters are:

- `position`: is the desired position of the figure (see \begin{figure}[position]). It could be `t` (top of the page), `b` (bottom of the page), `h` (at the command location if possible) or `H` (at command location);

Figure 5.3: Example of figure with source text

This figure illustrates the rendering of the explanation and a source text that are attached to a figure

Source:This is the source of the figure

- `include_graphics_options`: are the options passed to graphic inclusion command. These options may be one of the followings:
    - `pgf[=true|false]` : if it is evaluated to true, use the PGF commands `\pgfdeclareimage` and `\pgfuseimage` for including the image; Otherwise use the command `\includegraphics` for including the image;

  By default, the command `\includegraphics` is used. Depending on the use of the previous options, the rest of the options in `include_graphics_options` are those for `\includegraphics` or for `\pgfdeclareimage`;
- `filename`: is the filename passed to `\includegraphics`;
- `caption`: is the caption of the figure (see \caption{caption});
- `label`: is the label used to reference the figure (see \label{fig:label}).
- `source text`: is the optinal text used to describe the source of the figure. If this value is provided, it is rendered below the caption itself.

5.4.2 Commands for Referencing Figures

Because the two commands above register a label with string starting with `fig:`, we propose the following function to easily access to the figure's references:

- `\figref{label}`: is equivalent to `\ref{fig:label}`;
- `\figpageref{label}`: is equivalent to `\pageref{fig:label}`.

5.4.3 Example without Explanation and Source Text

The figure 5.4 page 40 is obtained with the command:

```
\mfigure[ht]{width=.4\linewidth}{slogo}{Example of figure inclusion with
\texttt{{\textbackslash}mfigure}}{example:mfigure1}
```

The reference and page reference are obtained with `\figref{example:mfigure1}` and `\figpageref{example:mfigure1}`.

Figure 5.4: Example of figure inclusion with `\mfigure`

5.4.4 Example with Source Text Only

For illustrating the rendering of the source text argument, the figure 5.5 page 41 is obtained with the command:

```
\mfigure[ht]{width=.4\linewidth}{slogo}{Example of figure inclusion with
\texttt{{\textbackslash}mfigure with source text}}{example:mfigure2}[This text
explain the source]
```

5.4.5 Rendering of the source text

The source text that could be provided to the commands `\mfigure` or `\mfigure*` is rendered with a specific configuration.

First the color of the source text is defined by `floatsourceforeground`. The default color is gray.

The prefix text that is written before the provided source text is defined by the command `\floatsourcename`.

It is also possible to change the color and the font of the source text by redefining the command `\mfigureformatsource[1]`. This command takes one argument that is the source text provided to `\mfigure` or `\mfigure*`.

5.5 Sub-figures

In some case, it is useful to put several images inside the same floating figure, but without loosing the possibility to reference each of the subfigures. This feature was proposed by the package `subcaption`. The following environments provides helper functions for `subcaption`.

Syntax 1.

```
\begin{mfigures}[position]{caption}{label}
...
\end{mfigures}
```

Syntax 2.

```
\begin{mfigures*}[position]{caption}{label}
```

Figure 5.5: Example of figure inclusion with `\mfigure with source text`

Source:This text explain the source

```
...
\end{mfigures*}
```

These two commands enable you to include an image in your document. The parameters are:

- `position`: is the desired position of the figure (see \begin{figure}[position]). It could be `t` (top of the page), `b` (bottom of the page), `h` (at the command location if possible) or `H` (at command location);
- `caption`: is the caption of the figure (see \caption{caption});
- `label`: is the label used to reference the figure (see \label{fig:label}).

Inside the environment `\mfigures[*]`, you could use the command `\mfigure` to properly include a subfigure (the first optional parameter is ignored), or you could use the command `\msubfigure{options}{file}{caption}`.

The figure 5.6 page 42 is obtained with the environment:

```
\begin{mfigures}{Example of subfigures with \texttt{msubfigures}}{example:msubfigure}
\mfigure{width=.4\linewidth}{img1}{First subfigure}{example:firstsubfigure}
\hspace{1cm}
\msubfigure{width=.4\linewidth}{img2}{Second subfigure}
\end{mfigures}
```

The reference and page reference are obtained with `\figref{example:msubfigure}` and `\figpageref{example:msubfigure}`.

The references to the subfigures could be obtained in two way:

- using the label given as the last parameter of `\mfigure`, eg. the label `example:firstsubfigure` corresponds to 5.8a;
- using the label of the enclosing figure to which the index of the subfigure could be appended (in its Roman representation and prefixed by the character "`:`"), eg. the label `example:msubfigure:b` corresponds to 5.6b;

5.5.1 Add a Figures Note

A figures note is a note that is rendered below the figures and its caption. This command changes the text of the figures note.

Syntax. `\figurenote{text}`

ARAKHNE.ORG
Another Yet Free Software Gateway
Example of a figure combined
A8-132
F/RF-111C/G
GUNSHIP GREY/OLIVE DRAB/CHARCOAL
Distribution of this artwork as part of the xfig package, where xfig is part of a commercially sold software package is permitted.
The author is not responsible for any losses which may result from the use or misuse of this material.
© 1995, Carlo Kopp

(a) First subfigure (b) Second subfigure

Figure 5.6: Example of subfigures with `mfigures`

Example. The figure 5.7 is an illustration of the following LaTeX code:

```
\begin{mfigures}{Example of subfigures with \texttt{msubfigures}}{example:msubfigure}
\figurenote{This is the text of the note}
\mfigure{width=.4\linewidth}{img1}{First subfigure}{example:firstsubfigure}
\hspace{1cm}
\msubfigure{width=.4\linewidth}{img2}{Second subfigure}
\end{mfigures}
```

This command has no effect when the class option `figurecaptionabove` is provided

5.5.1.1 — Add a Figures Source

A figures source is a text that is describing the source of the content of the figures.

Syntax. `\addsource{text}`

Example. The figure 5.8 is an illustration of the following LaTeX code:

```
\begin{mfigures}{Example of subfigures with \texttt{msubfigures}}{example:msubfigure}
\addsource{This is the text of the source}
\mfigure{width=.4\linewidth}{img1}{First subfigure}{example:firstsubfigure}
\hspace{1cm}
\msubfigure{width=.4\linewidth}{img2}{Second subfigure}
\end{mfigures}
```

This command has no effect when the class option `figurecaptionabove` is provided

5.6 Figures with embedded TEX commands

In several cases it is useful to include TEX commands inside a figure. It is possible to combine figures and TEX commands. Several figure editors provide exporting features to obtain combined figures: `xfig`, `inkscape`, `GNU Plot`, etc. Basically, these tools create two files per source figure:

ARAKHNE.ORG
Another Yet Free Software Gateway
Example of a figure combined
A8-132
F/RF-111C/G
GUNSHIP GREY/OLIVE DRAB/CHARCOAL
Distribution of this artwork as part of the xfig package, where xfig is part of a commercially sold software package is permitted.
The author is not responsible for any losses which may result from the use or misuse of this material.
© 1995, Carlo Kopp

(a) First subfigure (b) Second subfigure

Figure 5.7: Example of subfigures with `mfigures` and a figures note
This is the text of the note

- the figure in PDF or Postscript format (filename extensions, `.pdf` or `.ps`); and
- a TEX file that contains the commands to put over the figure, and that is including the generated figure. Its filename extension depends on the type of the figure: `.pdftex_t` or `.pdf_tex` for PDF, and `.pstex_t` or `.ps_tex` for Postscript.

To include this combined figure in your document, you simply need to include the generated TEX file (see below for details).

5.6.1 Include a Combined Picture/TEX Figure

To include a figure with TEX commands inside, you must have:

1. a Postcript figure (`.eps`), and a TEX file `.pstex_t` related to the Postscript figure; or
2. a PDF figure (`.pdf`), and a TEX file `.pdftex_t` related to the PDF figure.

With the `upmethodology-fmt` package, the inclusion of the figure with embedded TEX commands is similar to the inclusion of figures with `\includegraphics`. You must type the following command.

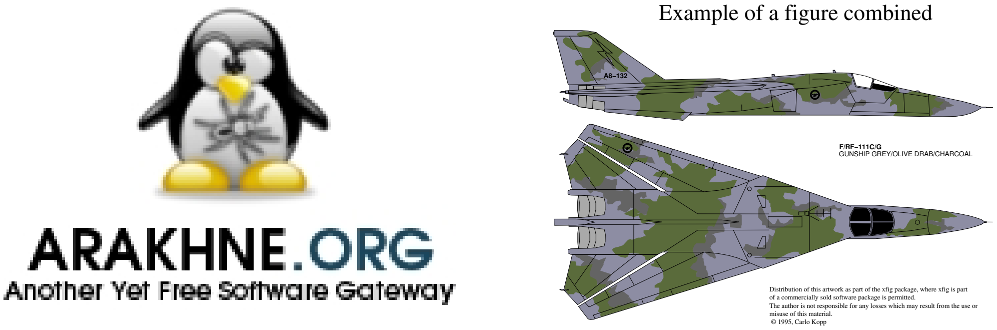
Syntax. `\includegraphicswtex[options]{filename}`

where `options` must be one or more of:

- `width=xxx`: specification of the width of the figure (`xxx` must be replaced by the length);
- `height=xxx`: specification of the height of the figure (`xxx` must be replaced by the length);

If the `filename` given to the command `\includegraphicswtex` does not specify a filename extension, the command tries to add the extensions `.pdftex_t`, `.pstex_t`, `.pdf_tex`, or `.ps_tex`, by default. If you want to specify other file extensions, you must use the command.

Syntax. `\DeclareGraphicsExtensionsWtex{extensions}`
where the `extensions` is a list of file extensions (including the point character), separated by coma characters.


ARAKHNE.ORG
Another Yet Free Software Gateway
Example of a figure combined
A8-132
F/RF-111C/G
GUNSHIP GREY/OLIVE DRAB/CHARCOAL
Distribution of this artwork as part of the xfig package, where xfig is part of a commercially sold software package is permitted.
The author is not responsible for any losses which may result from the use or misuse of this material.
© 1995, Carlo Kopp

(a) First subfigure (b) Second subfigure

Figure 5.8: Example of subfigures with `mfigures` and a figures source

Source:This is the text of the source

Example. `\DeclareGraphicsExtensionsWtex{.pdftex,.pstex}`

If the `filename` does not correspond to a file on the disk, the command `\includegraphicswtex` tries to find the file in the directories specified in `\graphicspath` (declared in the package `graphicx` for example).

Example. `\graphicspath{{./imgs/},{./imgs/additional/}}`

Note that each of the given directories must be finished by the separation character of your operating system: `/` on Unix, \ on Windows. You must always use the Unix standard because it is assumed by a lot of TEX compilers, even on Windows platforms.

Figure 5.9 gives an example of a floating figure combined with TEX commands, which is using the command `\includegraphicswtex`.

5.6.2 Floating figure with embedded TEX commands

To put a floating figure with TEX command inside, you may use one of the commands.

Syntax 1. `\mfigurewtex[position]{include_graphics_options}{filename}{caption}{label}`

Syntax 2. `\mfigurewtex*[position]{include_graphics_options}{filename}{caption}{label}`

The parameters are:

- `position`: is the desired position of the figure (see \beginfigure[position]). It could be `t` (top of the page), `b` (bottom of the page), `h` (at the command location if possible) or `H` (at command location);
- `include_graphics_options`: are the options to pass to `\includegraphicswtex`. For ascendant compatibility, if you pass a length without a key, e.g. `{.8\linewidth}`, the length is assumed to be the width of the figure;
- `filename`: is the name of the file of the figure (see `\includegraphicswtex` for details);
- `caption`: is the caption of the figure (see \caption{caption});
- `label`: is the label used to reference the figure (see \label{fig:label}).

The difference between `\mfigurewtex` and `\mfigurewtex*` is the same as the difference between `\begin{figure}` and `\begin{figure*}`: the star-version fits to the entire paper width event if the document has two or more columns.

Because the two commands above register a label with string starting with `fig:`, the commands `\figref` and `\figpageref` could be used.

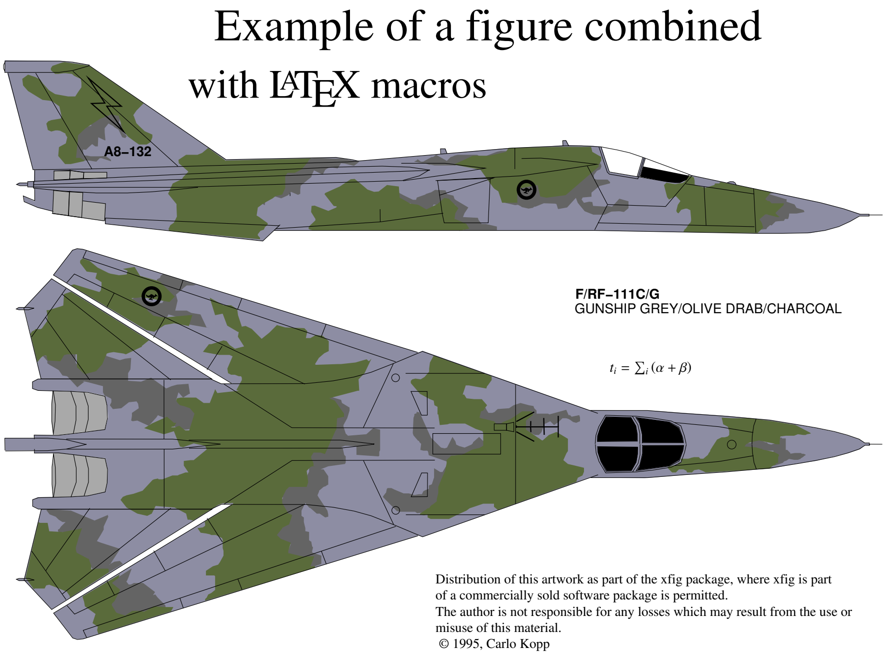

Example of a figure combined
with LaTeX macros
A8–132
F/RF–111C/G
GUNSHIP GREY/OLIVE DRAB/CHARCOAL
$t_i = \sum_i (\alpha + \beta)$
Distribution of this artwork as part of the xfig package, where xfig is part
of a commercially sold software package is permitted.
The author is not responsible for any losses which may result from the use or
misuse of this material.
© 1995, Carlo Kopp

Figure 5.9: Example of a figure combined with TeX commands

Figure 5.9 gives an example of a floating figure combined with TeX commands. Note that:

- the title of the figure contains the command `\LaTeX`, which produces: LaTeX;
- a small equation, written in TeX, is put between the two planes;

5.6.3 Helpers for embedded TeX

To help you to put TeX commands in a figure, and to define its real test inside the LaTeX document, several functions are provided:

- `\figmath{id}{expr}` will associate to the given identifier the given mathematical expression,
- `\figtext{id}{expr}` will associate to the given identifier the given text expression;

These expressions, defined with the two previous functions, may be referenced in the figure by a TeX command with a name similar to `\FIG`δ, where δ must be replaced by an identifier of your choice and used as parameter of one of the two previous functions (example: `\FIGmyid`).

Figure 5.9 gives an example where the equation is written as: `\FIGexampleofexpression` in the figure, and it is replaced by the real equation with:
`\figmath{exampleofexpression}{t_i = \sum_i \left(\alpha + \beta\right)}`

5.7 Tabulars

You could include a tabular (nor not-floating table) inside your document with the following environment. The UPmethodology provides two types of tabular environments: regular tabular, long tabular and table. Table 5.1 provides a brief comparison of these three concepts, that are detailed in sections 5.7.1, 5.7.2 and 5.8, respectivelly.

Criterion	*Regular Tabular*	*Long Tabular*	*Table*
Purpose	Creating the structure of a table	Same as regular tabular	Used as a floating container for tabular material
Floating Behavior	No	No	Yes
Placement Control	Where the tabular is written	Where the tabular is written	LaTeX decides according to provided position modifiers
Multiple pages	No	Yes	No

Table 5.1: Comparison of the concepts of regular tabular, long tabular and table

5.7.1 Regular Tabular

The regular tabular is designed to create tables by organizing data into rows and columns. According to Table 5.1, a regular tabular cannot cover multiple pages.

Syntax. `\begin{mtabular}[width]{ncolumns}{columns}{top title}...\end{mtabular}`

This tabular is an extension of the `xltabular` environment which provides dynamic columns with the specifier `X`, mixing the `lontable` and `tabularx` standard packages. The parameters are:

- `width`: is the desired width of the tabular;
- `ncolumns`: is the count of columns in the tabular. It must be consistent with the column description;
- `columns`: is the description of the columns according to the `tabular` and `tabularx` or `xltabular` packages. The following characters are supported for representing a single column :
  - `l`: the column is left aligned with a width that is corresponding to the maximum width of the column's content;
  - `r`: the column is right aligned with a width that is corresponding to the maximum width of the column's content;
  - `c`: the column is centered with a width that is corresponding to the maximum width of the column's content;
  - `p{w}`: the column is centered with a width equals to `w`;
  - `X`: the column is centered with a width computed by LaTeX.
- `top title`: is the text to be drawn as a title at the top of the table. This top title is not a caption, according to the definition of a caption in a table.

The number of columns must be the same for `ncolumns` and `columns`.

It is recommended to use as least one `X` in the specification of the `columns`.

The `mtabular` environment provides:

- `\tabulartitleinside{title}`
  This command allows you to define the title of the tabular. It uses the colors `tableheaderbackground` and `tableheaderforeground` for the background and the foreground respectively. The title has two lines at the top, and a single line below;

- `\tabularheader{`*header*$_1$`}...{`*header*$_n$`}`
  This command allows you to define the titles of the columns. It uses the colors `tableheaderbackground` and `tableheaderforeground` for the background and the foreground respectively. Because the count of columns was given to the environment this function takes the same count of parameters as the count of columns. This command adds a line after the header, *BUT NOT BEFORE.*

  The header text *header*$_1$ to *header*$_n$ ould have multiple lines. It is allowed to use the command \\ to add a line in the column's title.

- `\tabularrowheader{title}`
  This command is designed to be used in the first cell of a row. It is rendering the cell as a row's header. A row header is a cell that is an header for the row. Only the row header cell has the header background color.

- `\tabulartitlespec{column_spec}`
  This command defines the specification of the column used to render the title of the table. The default value of the column specification is |c|.

The following command is deprecated: `\tabulartitle`.

The following example of table is obtained by:

```
\begin{mtabular}[\linewidth]{4}{lXrX}{}
 \tabularheader{Col1}{Col2}{Col3}{Col4}
 a & b & c & d \\
 \hline
 e & f & g & h \\
 \tabulartitleinside{Example of second title in the table}
 \hline
 \tabularrowheader{i} & j & k & l \\
 \tabularheader{Col1-2}{Col2-2}{Col3-2}{Col4-2}
 m & n & o & p \\
\end{mtabular}
```

Col1	*Col2*	*Col3*	*Col4*
a	b	c	d
e	f	g	h
Example of second title in the table			
i	j	k	l
Col1-2	*Col2-2*	*Col3-2*	*Col4-2*
m	n	o	p

5.7.2 Long Tabular

Long tables could be rendered on multiple pages, according to Table 5.1, page 46. The `mlongtabular` environment supports long tables.

Syntax. `\begin{mlongtabular}[width]{ncolumns}{columns}{top title}...\end{mlongtabular}`

Example. An example of a long table could be found below.

This is the long table title

Col1	*Col2*
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	1
11	1
12	1
13	1
14	1
15	1
16	1
17	1
18	1
20	2
21	2
22	2
23	2
24	2
25	2
26	2
27	2
28	2
29	2

30	3
31	3
32	3
33	3
34	3
35	3
36	3
37	3
38	3
39	3
40	4
41	4
42	4
43	4
44	4
45	4
46	4
47	4
48	4
49	4
50	5
51	5
52	5
53	5
54	5
55	5
56	5
57	5
58	5
59	5
60	6
61	6
62	6
63	6
64	6
65	6
66	6
67	6
68	6
68	6

5.8 Tables

You could include a table inside your document with the following environment:
`\begin{mtable}[options]{width}{ncolumns}{columns}{caption}{label}...\end{mtable}`

According to Table 5.1, page 46, a table is a floating structure that contains data formatted in rows and columns.

This environment is based on the `mtabular` environment (see Section 5.7). The parameters are:

- `options`: are the options to pass to the `mtable` environment:
    - a table placement composed of one or more of the following characters. The order in which the placement options are specified does not make any difference, as the placement options are always attempted in the order `h-t-b-p`. Thus `[hb]` and `[bh]` are both attempted as `h-b`. The more float placement options are given to LaTeX, the better it handles float placement. Consequently, and because we want a simple TeX code in the background, all the permutations are not supported by the `mtable` environment. We recommend to put placement letters in the order they appear in the following list:
        - `h`: Place the float here, i.e., approximately at the same point it occurs in the source text (however, not exactly at the spot),
        - `t`: Position at the top of the page,
        - `b`: Position at the bottom of the page,
        - `p`: Put on a special page for floats only,
        - `H`: Places the float at precisely the location in the LaTeX code. Requires the `float` package. This is somewhat equivalent to `h!`;
        - `!`: Override internal parameters LaTeX uses for determining "good" float positions,

      If you specify more than one table placement in the options, the last one is used.
    - `size=<command>`: specify the size of the text in the table (by default, `\normalsize`);
- `width`: is the desired width of the table (ie., the tabular inside the table);
- `ncolumns`: is the count of columns in the table (ie., the tabular inside the table). It must be consistent with the column description;
- `columns`: is the description of the columns according to the `tabular` and `tabularx` and `xltabular` packages;
- `caption`: is the caption of the table;
- `label`: is the label referencing the table.

⚠ The number of columns must be the same for `ncolumns` and `columns`.

✎ It is recommended to use as least one `X` in the specification of the `columns`.

Because the `mtable` environment registers a label with a string starting with `tab:`, the following functions are proposed to easily access to the table's references:

- `\tabref{label}`: is equivalent to `\ref{tab:label}`;
- `\tabpageref{label}`: is equivalent to `\pageref{tab:label}`.

The table 5.2 page 51 is an illustration of the following LaTeX code:

```
\begin{mtable}{\linewidth}{4}{lXrX}{Example of \texttmtable}{example:mtable}
\tabularheader{Col1}{Col2}{Col3}{Col4}
a & b & c & d \\
\hline
e & f & g & h \\
\end{mtable}
```

Col1	*Col2*	*Col3*	*Col4*
a	b	c	d
e	f	g	h

Table 5.2: Example of `mtable`

5.8.1 Table Components

The package provides several commands that could help you to configure, update or add a component ot the table.

5.8.1.1 — Add a Table Note

A table note is a note that is rendered below the table and its caption. This command changes the text of the table note.

Syntax. `\tablenote{text}`

Col1	*Col2*	*Col3*	*Col4*
a	b	c	d
e	f	g	h

Table 5.3: Example of `mtable` with a table note
This is the text of the note

Example. The table 5.3 is an illustration of the following LaTeX code:

```
\begin{mtable}{\linewidth}{4}{lXrX}{Example of \texttmtable with a table
note}{example:mtable2}
\tablenote{This is the text of the note}
\tabularheader{Col1}{Col2}{Col3}{Col4}
a & b & c & d \\
\hline
e & f & g & h \\
\end{mtable}
```

⚠ This command has no effect when the class option `tablecaptionabove` is provided

5.8.1.2 — Add a Table Source

A table source is a text that is describing the source of the content of the table.

Syntax. `\addsource{text}`

Example. The table 5.4 is an illustration of the following LaTeX code:

```
\begin{mtable}{\linewidth}{4}{lXrX}{Example of \texttmtable with a table
source}{example:mtable3}
\addsource{This is the source of the content}
\tabularheader{Col1}{Col2}{Col3}{Col4}
a & b & c & d \\
\hline
e & f & g & h \\
```

Col1	*Col2*	*Col3*	*Col4*
a	b	c	d
e	f	g	h

Table 5.4: Example of `mtable` with a table source

Source:This is the source of the content

> This command has no effect when the class option `tablecaptionabove` is provided

5.9 Enumerations

The package `upmethodology-fmt` provides a set of commands dedicated to enumeration lists.

5.9.1 Enumeration Counters

Sometimes it is useful to start an enumeration list from a specific given number. This package provides several commands for saving and restoring the counter use by the enumeration lists.

> Only one counter could be saved at a given time. It means that you cannot save the counters for an enumeration and for an enclosing enumeration at the same time.

Two general commands are defined for helping you to save a counter value into the global variable:

- `\savecounter{name}`
  save the value of the counter identified by the given name in a global variable. The name of the counter must be previously defined with one of the standard LaTeX or TeX commands, e.g. `\newcounter`;
- `\restorecounter{name}`
  put the previously saved value into the counter with the given name. The name of the counter must be previously defined with one of the standard LaTeX or TeX commands, e.g. `\newcounter`;

The counter is extensively used in enumeration lists. The following commands will help you for managing the enumeration counter:

- `\setenumcounter{value}`
  force the value of the counter used by the enumeration environments;
- `\getenumcounter`
  replies the value of the counter used by the enumeration environments;
- `\saveenumcounter`
  save the value of the counter used by the enumeration environment with `\savecounter`;
- `\restoreenumcounter`
  restore the value of the counter used by the enumeration environment with `\restorecounter`.

Example:. The following LaTeX code produces the result below:

```
This is a text: \begin{enumerate}
        \item This is an item.
        \item This is another item.
        \saveenumcounter
\end{enumerate}
This is a text in the between.
\begin{enumerate}
        \restoreenumcounter
        \item The list goes on
        \item and on.
\end{enumerate}
This is a second text in the between.
\begin{enumerate}
        \setenumcounter{18}
        \item The list goes on again
        \item and on.
\end{enumerate} This is the text after.
```

This is a text:

1. This is an item.

2. This is another item.

This is a text in the between.

3. The list goes on

4. and on.

This is a second text in the between.

18. The list goes on again

19. and on.

This is the text after.

5.9.2 Inline Enumeration

In several document, an enumeration of things is written inside a paragraph instead of inside a list of points.

5.9.2.1 — Standard inline enumeration from `enumitem`

The UPmethodology package includes the package `enumitem` with the `inline` option. Therefore, it is possible to obtain an inline (or horizontal) enumeration.

Example:. The following LaTeX code produces the result below:

```
This is a text: \begin{enumerate*}
\item first thing;
\item second thing;
\item etc.
\end{enumerate*} This is the text after.
```

This is a text: 1. first thing; 2. second thing; 3. etc. This is the text after.

5.9.2.2 — Shortcut inline enumeration

The environment `inlineenumeration` is a shortcut to `enumerate*` with the options `label=\textit(\roman*),ref=\textit(\roman*)`.

Example:. The following LaTeX code produces the result below:

```
This is a text: \begin{inlineenumeration}
\item first thing;
\item second thing;
\item etc.
\end{inlineenumeration} This is the text after.
```

This is a text: *(i)* first thing; *(ii)* second thing; *(iii)* etc.This is the text after.

5.10 Environment description

The environment `description` is redefined as following.

Syntax.

```
\begin{description}[separator]
\item[desc] text
\end{description}
```

The text put in place of `desc` represents the text which may be emphasized in the description item. The `separator` is the text that is inserted at the end of the head of each description item.

Example 1. The following LaTeX code, using Roman numbers, produces the description just below:

```
\begin{description}
\item[first thing] this is a text for the first thing;
\item[second thing] this is a text for the second thing;
\item[more] etc.
\end{description}
```

- **first thing:** this is a text for the first thing;
- **second thing:** this is a text for the second thing;
- **more:** etc.

Example 2. The following LaTeX code produces the description just below:

```
\begin{description}[///]
\item[first thing] this is a text for the first thing;
\item[second thing] this is a text for the second thing;
\item[more] etc.
\end{description}
```

- **first thing///** this is a text for the first thing;
- **second thing///** this is a text for the second thing;
- **more///** etc.

5.11 Descriptions in conjunction with enumeration

It may be helpful to put a list of descriptions in conjunction with an enumeration. In other words, the following environment provides a mix between the standards LaTeX environments `description` and `enumerate`.

5.11.1 Environment `enumdescription`

The environment `enumdescription` is:

Syntax.

```
\begin{enumdescription}[type]
\item[desc] text
\end{enumdescription}
```

where the `type` is the type of the enumeration. It may be one of:

- “`i`”: for an enumeration with Roman numbers (this is the default),
- “`1`”: for an enumeration with Arabic numbers,
- “`a`”: for an enumeration with letters.

The text put in place of `desc` represents the text which may be emphasized in the description item.

To change the rendering of the labels, you must redefined the command as:

```
\renewcommand{\enumdescriptionlabel}[1]{ ... #1 ... }
```

To change the separator between the counter and the description, you must redefined the command as:

```
\renewcommand{\enumdescriptioncounterseparator}{ ... }
```

To change the separator between the description and the rest of the text, you must redefined the command as:

```
\renewcommand{\enumdescriptionlabelseparator}{ ... }
```

Example 1. The following LaTeX code, using Roman numbers, produces the enumerated description just below:

```
\begin{enumdescription}
\item[first thing] this is a text for the first thing;
\item[second thing] this is a text for the second thing;
\item[more] etc.
\end{enumdescription}
```

i - first thing: this is a text for the first thing;

ii - second thing: this is a text for the second thing;

iii - more: etc.

Example 2. The following LaTeX code, using numeric numbers, produces the enumerated description just below:

```
\begin{enumdescription}[1]
\item[first thing] this is a text for the first thing;
\item[second thing] this is a text for the second thing;
\item[more] etc.
\end{enumdescription}
```

1 - first thing: this is a text for the first thing;

2 - second thing: this is a text for the second thing;

3 - more: etc.

Example 3. The following LaTeX code, using letter numbers, produces the enumerated description just below:

```
\begin{enumdescription}[a]
\item[first thing] this is a text for the first thing;
\item[second thing] this is a text for the second thing;
\item[more] etc.
\end{enumdescription}
```

a - first thing: this is a text for the first thing;

b - second thing: this is a text for the second thing;

c - more: etc.

5.11.2 Environment `enumerate`

The environment `enumerate` exists in the standard LaTeX distributions. The UPM package redefines this environment to provide a behavior similar to the one of the environment `enumdescription`.

Additionally, you could specify the format of the counter in the first optional parameter. This format is a text in which the first occurrence of one of the following characters is replaced by the value of the counter with the associated number format:

- `1`: the counter is an arabic number;
- `a`: the counter is a sequence of lower-case alphabetic letters;
- `A`: the counter is a sequence of upper-case alphabetic letters;
- `i`: the counter is a lower-case roman number;
- `I`: the counter is an upper-case roman number.

Example 1. The following LaTeX code produces a list, which is similar to the one generated by the standard LaTeX environment `enumerate`:

```
\begin{enumerate}
\item this is a text for the first thing;
\item this is a text for the second thing;
\item etc.
\end{enumerate}
```

1. this is a text for the first thing;

2. this is a text for the second thing;

3. etc.

Example 2. The following LaTeX code illustrates how the environment is reacting to a given description:

```
\begin{enumerate}
\item this is a text for the first thing;
\item[description] this is a text for the second thing;
\item etc.
\end{enumerate}
```

1. this is a text for the first thing;

2. description: this is a text for the second thing;

3. etc.

Example 3. The following LaTeX code illustrates the alphabetic counter specification. Note that the parenthesis characters are directly rendered in the list:

```
\begin{enumerate}[(a)]
\item this is a text for the first thing;
\item this is a text for the second thing;
\item etc.
\end{enumerate}
```

(a) this is a text for the first thing;

(b) this is a text for the second thing;

(c) etc.

Example 4. The following LaTeX code illustrates the roman counter specification. Note that the dot character is directly rendered in the list:

```
\begin{enumerate}[I.]
\item this is a text for the first thing;
\item this is a text for the second thing;
\item etc.
\end{enumerate}
```

I. this is a text for the first thing;

II. this is a text for the second thing;

III. etc.

5.11.3 Environment `enumdescriptionx`

The environment `enumdescriptionx` extends the environment `enumdescription` by enabling a finer configuration with more parameters.

Syntax.

```
\begin{enumdescriptionx}[type]{counter\_prefix}{counter\_postfix}
\item[desc] text
\end{enumdescriptionx}
```

where the `type` is the type of the enumeration. It may be one of:

- "`i`": for an enumeration with Roman numbers (this is the default),
- "`1`": for an enumeration with Arabic numbers,
- "`a`": for an enumeration with letters.

The text put in place of `desc` represents the text which may be emphasized in the description item. The text `counter_prefix` is put before all the counter values in the enumeration. The text `counter_postfix` is put after all the counter values in the enumeration.

To change the rendering of the labels, you must redefined the command as:

```
\renewcommand{\enumdescriptionlabel}[1]{ ... #1 ... }
```

Example. The following LaTeX code, using letter numbers, produces the enumerated description just below:

```
\begin{enumdescriptionx}[a]{$\langle$}{$\rangle$}
\item[first thing] this is a text for the first thing;
\item[second thing] this is a text for the second thing;
\item[more] etc.
\end{enumdescriptionx}
```

⟨a⟩ - first thing: this is a text for the first thing;

⟨b⟩ - second thing: this is a text for the second thing;

⟨c⟩ - more: etc.

5.12 Footnotes

The package `upmethodology-fmt` provides a set of commands allowing to save the reference number of a footnote and to recall this reference many time as required.

- `\savefootnote{footnote text}{footnote id}`
  put a footnote and mark it with the corresponding label.
  Example: `\savefootnote{This is an example of a recallable footnote}{footrecalla}`[1];

---

[1] This is an example of a recallable footnote

- `\savefootnote*{footnote text}{footnote id}`
  mark a footnote with the corresponding label but do not put in the current page.
  Example 1: `\savefootnote*{This is a second example of a recallable footnote}{footrecallb}`;
  Example 2: `\savefootnote*{This is a third example of a recallable footnote}{footrecallc}`.

- `\reffootnote{footnote id}`
  recall the footnote reference without page number.
  Example 1: `\reffootnote{footrecalla}`[1] = B;
  example 2: `\reffootnote{footrecallb}`[2] = A;
  example 4: `\reffootnote{footrecalld}`[??] =?.

- `\reffootnote*{footnote id}`
  recall the footnote reference with the page number if different of the current page.
  Example 1: `\reffootnote*{footrecalla}`[1(58)];
  example 2: `\reffootnote*{footrecallb}`[2(59)];
  example 3: `\reffootnote*{footrecallc}`[3];
  example 4: `\reffootnote*{footrecalld}`[??(??)].

5.13 UML diagrams on the side of paragraphs

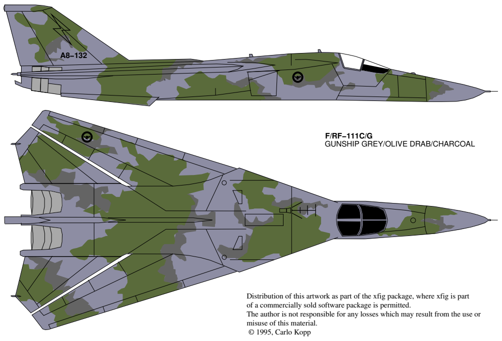
The package `upmethodology-fmt` provides an environment that makes it possible to put an UML diagram (or any other picture) on the side of a paragraph.

- `\begin{umlinpar}[width]{picture_path}`
  `text`
  `\end{umlinpat}`
  put the specified picture on the side of the given text. The optional parameter `width` corresponds to the desired width ofthe picture. By default it is `.5\linewidth`.

This paragraph is an typical example of the usage of the environment `umlinpar`. To obtain it, the following LaTeX code was typed:

```
\begin{umlinpar}{smalllogo}
This paragraph is an typical example
of the usage of the environment
\texttt{umlinpar}.
\end{umlinpar}
```

Example of a figure combined


A8–132
F/RF–111C/G
GUNSHIP GREY/OLIVE DRAB/CHARCOAL
Distribution of this artwork as part of the xfig package, where xfig is part
of a commercially sold software package is permitted.
The author is not responsible for any losses which may result from the use or
misuse of this material.
© 1995, Carlo Kopp

5.14 Date formatting

Because the concept of date was important and unfortunately localized, this package provides a set of functions to define and extract information from dates (the supported date formats are described in table 5.5):

---

[2] This is a second example of a recallable footnote

[3] This is a third example of a recallable footnote

- `\makedate{day}{month}{year}`
  allows you to create the text corresponding to the given date according to the current localized date format.
- `\extractyear{formatted_date}`
  extract the year field from a date respecting the localized date format.
- `\extractmonth{formatted_date}`
  extract the month field from a date respecting the localized date format.
- `\extractday{formatted_date}`
  extract the day field from a date respecting the localized date format.

`yyyy/mm/dd`	default format
`dd/mm/yyyy`	french format

Table 5.5: List of supported date formats

5.15 Text formatting

The package `upmethodology-fmt` provides a set of commands to format the text.

- `\textsup{text}`
  put a text as exponent in text mode instead of the basic LaTeX exponent in math mode. In opposite to the standard LaTeX command `\textsuperscript`, this command adds an extra space after the command when needed.
  Example: `\textsup{this is an exponent}`$^{\text{this is an exponent}}$ this is the following text;
- `\textup{text}`
  same as `\textsup`.
- `\textsub{text}`
  put a text as indice in text mode instead of the basic LaTeX indice in math mode. In opposite to `\textsubscript`, this command adds an extra space after the command when needed. In opposite to `\textdown`, the size of the text is not changed in the text down.
  Example: `\textsub{this is an indice}`$_{\text{this is an indice}}$ this is the following text;
- `\textdown{text}`
  put a text as indice in text mode instead of the basic LaTeX indice in math mode. In opposite to `\textsubscript`, this command adds an extra space after the command when needed. In opposite to `\textsub`, the size of the text is changed in the text down.
  Example: `\textdown{this is an indice}`$_{\text{this is an indice}}$ this is the following text;
- `\textsubscript{text}`
  put a text as indice in text mode instead of the basic LaTeX indice in math mode. As for the standard LaTeX command `\textsuperscript`, this command does not add an extra space after the command.
  Example: `\textsubscript{this is an indice}`$_{\text{this is an indice}}$this is the following text;
- `\Emph{text}`
  put a *very important* text. This command is similar to the standard LaTeX command `\emph`. The difference is: `\emph` is for "important things"; and `\Emph` is for "very important things".
  Example: This text is `\emph{important}`, but this one is `\Emph{very important}`
  gives: This text is *important*, but this one is **very important**;

- `\makename[von]{first name}{last name}`
  format the specified people name components according to the document standards. By default, the format `first von last` is used.
  Example: `\makename[von]{Ludwig Otto Frederik Wilhelm}{Wittelsbach}`,
  "LUDWIG OTTO FREDERIK WILHELM von WITTELSBACH";

- `\upmmakename[von]{first name}{last name}{separator}`
  format the specified people name components according to the document standards. By default, the format `first von last` is used.
  Example: `\upmmakename[von]{Ludwig Otto Frederik Wilhelm}{Wittelsbach}{/}`,
  "LUDWIG OTTO FREDERIK WILHELM/von/WITTELSBACH";

- `\makenamespacing{name}`
  format the specified name to be sure that the spaces after the points of the initials are demi-spaces.
  Example: `\makenamespacing{S.G.}Galland`,
  "S. G. Galland";

- `\makelastname{name}`
  format the specified last/family name.
  Example: `\makelastname{Galland}`,
  "GALLAND";

- `\makefirstname{name}`
  format the specified first name.
  Example: `\makefirstname{Stéphane}`,
  "STÉPHANE";

- `\prname[von]{first name}{last name}`
  `\prname*[von]{first name}{last name}`
  format the specified people name components according to the document standards for *Professor* title. By default, the format `first von last` is used. The star-ed version is post-fixed, the non-star-ed version is prefixed.
  Example 1: `\prname{Pierre}{Martin}`, "PR. PIERRE MARTIN";
  Example 2: `\prname*{Pierre}{Martin}`, "PIERRE MARTIN, PR.";

- `\drname[von]{first name}{last name}`
  `\drname*[von]{first name}{last name}`
  format the specified people name components according to the document standards for *Doctor* title. By default, the format `first von last` is used. The star-ed version is post-fixed, the non-star-ed version is prefixed.
  Example 1: `\drname{Pierre}{Martin}`, "DR. PIERRE MARTIN";
  Example 2: `\drname*{Pierre}{Martin}`, "PIERRE MARTIN, DR.";

- `\phdname[von]{first name}{last name}`
  `\phdname*[von]{first name}{last name}`
  format the specified people name components according to the document standards for *Philosophiæ-Doctor* title. By default, the format `first von last` is used. The star-ed version is post-fixed, the non-star-ed version is prefixed.
  Example 1: `\phdname{Pierre}{Martin}`, "PH.D. PIERRE MARTIN";
  Example 2: `\phdname*{Pierre}{Martin}`, "PIERRE MARTIN, PH.D.";

- `\scdname[von]{first name}{last name}`
  `\scdname*[von]{first name}{last name}`
  format the specified people name components according to the document standards for *ScientiæDoctor* title. By default, the format `first von last` is used. The star-ed version is post-fixed, the non-star-ed version is prefixed.
  Example 1: `\scdname{Pierre}{Martin}`, "SC.D. PIERRE MARTIN";
  Example 2: `\scdname*{Pierre}{Martin}`, "PIERRE MARTIN, SC.D.";

- `\mdname[von]{first name}{last name}`
  `\mdname*[von]{first name}{last name}`
  format the specified people name components according to the document standards for *Medicinæ-Doctor* title. By default, the format `first von last` is used. The star-ed version is post-fixed, the non-star-ed version is prefixed.
  Example 1: `\mdname{Pierre}{Martin}`, “M.D. Pierre Martin”;
  Example 2: `\mdname*{Pierre}{Martin}`, “Pierre Martin, M.D.”;

- `\pengname[von]{first name}{last name}`
  `\pengname*[von]{first name}{last name}`
  format the specified people name components according to the document standards for *Professional/Chartered Engineer* title. By default, the format `first von last` is used. The star-ed version is post-fixed, the non-star-ed version is prefixed.
  Example 1: `\pengname{Pierre}{Martin}`, “CEng. Pierre Martin”;
  Example 2: `\pengname*{Pierre}{Martin}`, “Pierre Martin, CEng.”;

- `\iengname[von]{first name}{last name}`
  `\iengname*[von]{first name}{last name}`
  format the specified people name components according to the document standards for *Incorporated Engineer* title. By default, the format `first von last` is used. The star-ed version is post-fixed, the non-star-ed version is prefixed.
  Example 1: `\iengname{Pierre}{Martin}`, “IEng. Pierre Martin”;
  Example 2: `\iengname*{Pierre}{Martin}`, “Pierre Martin, IEng.”.

5.16 Symbols

5.16.1 Symbols in Text Mode

The package `upmethodology-fmt` provides several symbols in text mode, and described inside the table 5.6.

`\arakhneorg`	Arakhnê.org
`\copyright`	©
`\trademark`	™
`\regmark`	®
`\smalltrade`	™
`\smallreg`	®
`\smallcopy`	©
`\ust`	$^{\text{st}}$
`\und`	$^{\text{nd}}$
`\urd`	$^{\text{rd}}$
`\uth`	$^{\text{th}}$

Table 5.6: List of symbols

5.16.2 Symbols in Math Mode

The package `upmethodology-fmt` provides several symbols in math mode, and described inside the table 5.7.

Sets	
`\R`	$\mathbb{R}$
`\N`	$\mathbb{N}$
`\Z`	$\mathbb{Z}$
`\Q`	$\mathbb{Q}$
`\C`	$\mathbb{C}$
`\powerset p`	$\mathcal{P}p$
Operators	
`\sgn expr`	$\operatorname{sgn} expr$

Table 5.7: List of symbols

5.17 Bibliography

The package `upmethodology-fmt` provides a set of commands allowing to manage the bibliography. The default bibliography style is `abbr`.

- `\bibliographystyle{style}`
  set the bibliography style to use.
  Example: `\bibliographystyle{alpha}`;
- `\bibliography{file}`
  set the BibTeX file to use.
  Example: `\bibliography{mybib}`;
- `\bibsize{size}`
  set the font size used for the bibliography section.
  Example: `\bibsize{\Huge}`;

5.18 Theorems and Mathematic Environments

The package `upmethodology-fmt` defines several environments and commands that are based on the `theorem` or the math API of LaTeX.

5.18.1 Definition of a new theorem environment

The theorem API from this package is based on the standard `tcolorbox` package. If you want to create a new theorem environment based on the style provided by this package, you could invoke `\declareupmtheorem`:

Syntax.

```
\declareupmtheorem[name of the tcolorbox style]{name of the LaTeX environment}{label
of the theorem}{title of the theorems' list}{prefix for the label keys}{name of the
theorem's list}{command for formating the source}
```

This command defines:

- the environment with the given ”name of the LaTeX environment”, and

- the command `\listof⟨name of the LaTeX environment⟩s`.

The `name of the tcolorbox style` is the name of the tcolorbox style to be used. This style must be defined according to the `tcolorbox` package documentation. By default, it is `upmdefinition`.

The `label of the theorem` is the text to put in the theorem header, e.g., `Definition`.

The `title of the theorems' list` is used by the command `\listof⟨name of the LaTeX environment⟩s` as the title of the associated chapter/section.

By default, the theorem environment defines a label for the theorem box. The `prefix for the label keys` is the prefix that is automatically appended to the label. For example, if this prefix is set to ”`prefix`”, and the theorem is invoked with a label value ”`thekey`”, then the theorem environment will automatically generate the command `\label{prefix:thekey}` in the theorem environment.

The `name of the theorems' list` is identifier of the list in which the theorem adds entries. This list could be displayed with the command `\listof⟨name of the LaTeX environment⟩s`.

The `command for formating the source` is LaTeX command that could be followed by the source text of theorem. It is usually used for formatting this source text in the lower part of the theorem box.

> ⚠ The command `\declareupmtheorem` can be used only inside the preamble of your document.

5.18.2 Using a defined theorem

After defining an environment, you could use it as a regular LaTeX environment, whitch takes three parameters:

- the title of the theorem in the box,
- the key for the implicit `\label` command, and
- the optional argument that is the text of the source. The source of the theorem is rendered in the lower part of the theorem box. Usually, it corecponds to the reference to a scientific article or an Internet page.

Example 1. The following code define the environment `mytheorem` and use it without source text:

```
\documentclass{upmethodology-document}
\declareupmtheorem{mytheorem}{My Theorem}{List of my Theorems}{thm}{mytheorem}{\textbf}
\begin{document}

Read the theorem \ref{thm:mytheo}.

\begin{mytheorem}{Theorem of Everything}{mytheo}
This is the theorem of Evereything.
\end{mytheorem}

\end{document}
```

gives the result:

Read the theorem 1.

My Theorem 1: Theorem of Everything

This is the theorem of Everything.

Example 2. The following code define the environment `mytheorem` and use it with a source text:

```
\documentclass{upmethodology-document}
\declareupmtheorem{mytheorem}{My Theorem}{List of my Theorems}{thm}{mytheorem}{\textbf}
\begin{document}

Read the theorem \ref{thm:mytheo2}.

\begin{mytheorem}{Theorem of Everything}{mytheo2}[description of the source]
This is the theorem of Evereything.
\end{mytheorem}

\end{document}
```

gives the result:

Read the theorem 2.

My Theorem 2: Theorem of Everything

This is the theorem of Everything.

description of the source

Reference the defined theorem. You could add a reference to a defined theorem by using one of the following commands:

- `\ref{⟨label of the theorem⟩:⟨Label of the theorem⟩}` for showing the definition's number, e.g., 2 for the theorem above.
- `\pageref{⟨label of the theorem⟩:⟨Label of the theorem⟩}` for showing the definition's page, e.g., 65 for the theorem above.

5.18.3 Using the list of theorems

When a theorem is defined, the it is possible to creation a chapter (in books or reports) or a section (in articles) that lists all the defined theorems. In order to do so, you could use the ”listof” command that is automatically created when defining the theorem. The following code define the environment `mytheorem` and shows the list of defined `mytheorem`s:

```
\documentclass{upmethodology-document}
\declareupmtheorem{mytheorem}{My Theorem}{List of my Theorems}{thm}{mytheorem}{\textbf}
\begin{document}

\begin{mytheorem}{Theorem of Everything}{mytheo}
This is the theorem of Evereything.
\end{mytheorem}
```

```
\listofmytheorems

\end{document}
```

5.18.4 `definition`

The package `upmethodology-fmt` defines the environment `definition` to put a definition in your document. This environment is based on the `theorem` environment explained in the previous sections. The `definition` takes one optional parameter: the name of the definition.

Example 1. The following LaTeX code:

```
\begin{definition}{Name of the definition}{Label of the definition}
   Text of the definition.
\end{definition}
```

produces:

> **Definition 1: Name of the definition**
>
> Text of the definition.

It is also possible to provide the source of the definition by adding the optional parameter as in the following example.

Example 2.

```
\begin{definition}{Name of the definition}{Label of the definition}[This is the source]
   Text of the definition.
\end{definition}
```

The result of this definition declaration is:

> **Definition 2: Name of the definition**
>
> Text of the definition.
>
> *Source: This is the source*

Reference the definition. You could add a reference to a definition by using one of the following commands:

- `\defref{⟨Label of the definition⟩}` for showing the definition's number, e.g., 3 for the definition above.
- `\defpageref{⟨Label of the definition⟩}` for showing the definition's page, e.g., 67 for the definition above.

Change the colors of the definition:. You could change the colors of the `definition` environment by redefining the colors below with one of the commands `\definecolor` or `\colorlet`:

- `definitionbackground` is the color of the background of the definition;

- `definitionborder` is the color of the frame and of the header's background;
- `definitionheaderforeground` is the color of the text in the header of the definition;
- `definitiontextforeground` is the color of the text in the body of the definition.
- `definitionsourceforeground` is the color of the text in the lower part of the definition in which the source is rendered.

Example of color redefinition:. The following LaTeX code:

```
\definecolor{definitionheaderforeground}{rgb}{.3,.5,.8}
\colorlet{definitionbackground}{gray!20}
\colorlet{definitionborder}{red}
\begin{definition}{Name of the definition}{Label of the definition}
  Text of the definition.
\end{definition}
```

produces:

Definition 3: Name of the definition

Text of the definition.

5.19 Algorithms

If you have included the LaTeX packages for algorithm floatings, e.g., `algorithm`, the default style for the algorithms may e altered with the following colors:

- `algorithmcaptionlabel`: is the color of the text for the label of the algorithms, e.g., `Algorithm 1:`;
- `algorithmcaption`: is the color of the text of the name of the description;
- `algorithmborder`: is the color of the rules that are drawn in the algorithm frame.

5.20 Emphazing Box

If you want to create a text that is emphazed with a box, you could use the environment:

Syntax. `\begin{emphbox}[width] text \end{emphbox}`

Example. The following LaTeX code:

```
\begin{emphbox}[.7\linewidth]
This is an emphazed text.
\end{emphbox}
```

produces:

This is an emphazed text.

Emphazing Box with a Title. Three additional boxes are provided. All of them are output a title, and has a different background color:

```
\begin{titleemphbox}[width]{title} text \end{titleemphbox}
\begin{titleemphbox2}[width]{title} text \end{titleemphbox2}
\begin{titleemphbox3}[width]{title} text \end{titleemphbox3}
```

The following LaTeX code:

```
\begin{titleemphbox}[.7\linewidth]{The title}
This is an emphazed text.
\end{titleemphbox}
```

produces:

The title

This is an emphazed text.

The following LaTeX code:

```
\begin{titleemphbox2}[.7\linewidth]{The title}
This is an emphazed text.
\end{titleemphbox2}
```

produces:

The title

This is an emphazed text.

The following LaTeX code:

```
\begin{titleemphbox3}[.7\linewidth]{The title}
This is an emphazed text.
\end{titleemphbox3}
```

produces:

The title

This is an emphazed text.

Change the colors of the emphazing box. You could change the colors of the `emphbox` environment by redefining the colors below with one of the commands `\definecolor` or `\colorlet`:

- `emphboxbackground` is the color of the background of the environment;
- `emphboxborder` is the color of the frame;
- `emphboxtext` is the color of the text in the body of the environment.
- `emphboxbackgroundb` is the color of the background of the second environment with title;
- `emphboxbackgroundc` is the color of the background of the third environment with title;

Example of color redefinition. The following LaTeX code:

```
\colorlet{emphboxbackground}{gray!20}
\colorlet{emphboxborder}{red}
\begin{emphbox}
This is an emphazed text.
\end{emphbox}
```

produces:

This is an emphazed text.

5.21 Framed Boxes or Mini Pages

There is two API that are available for created framed boxes. The first one is `tcolorbox`, that is the one use by default in this package. The second approach is based on the `framedminipage` commands defined in this package.

5.21.1 With tcolorbox

`tcolorbox` provides an environment for colored and framed text boxes with a heading line. Optionally, such a box can be split in an upper and a lower part. The package `tcolorbox` is included by default, with its `theorems` extension. It can be used for the setting of LaTeX examples where one part of the box displays the source code and the other part shows the output. Another common use case is the setting of theorems (see Section 5.18).

More information could be found on https://www.ctan.org/pkg/tcolorbox or https://github.com/T-F-S/tcolorbox.

5.21.2 With framedminipage

Standard LaTeX distribution provides the `minipage` environment. This environment allows you to put a small piece of page inside your document. The package `upmethodology-fmt` provides two framed extensions of the original `minipage` environment: `framedminipage` and `framedcolorminipage`.

The prototypes of there two new environments are, respectively:

- `\begin{framedminipage}{width} ...\end{framedminipage}`
- `\begin{framedcolorminipage}{width}{border_color}{background_color} ...\end{framedcolorminipage}`

Example of `framedminipage`. The following LaTeX code:

```
\begin{framedminipage}{.75\linewidth}
This is a text inside a framed minipage.
\end{framedminipage}
```

produces: This is a text inside a framed minipage.

Example of `framedcolorminipage`. The following LaTeX code:

```
\begin{framedcolorminipage}{.75\linewidth}{red}{yellow}
This is a text inside a framed minipage with colors.
\end{framedcolorminipage}
```

produces: This is a text inside a framed minipage with colors.

5.22 Message Boxes

The package `upmethodology-fmt` provides a set of environment to put emphasis message boxes in the text. Three types of boxes are supported: caution, information, and question.

```
\begin{upmcaution}[width]
This is an example of a caution
message.
\end{upmcaution}
```

⚠ | This is an example of a caution message.

```
\begin{upminfo}[width]
This is an example of an information
message.
\end{upminfo}
```

ℹ | This is an example of an information message.

```
\begin{upmnote}[width]
This is an example of a note message.
\end{upmnote}
```

✎ | This is an example of a note message.

```
\begin{upmquestion}[width]
This is an example of a question
message.
\end{upmquestion}
```

? | This is an example of a question message.

```
\begin{upmaction}[width]
This is an example of an action
message.
\end{upmaction}
```

▶ | This is an example of an action message.

5.23 Additional Commands for the Table of Content

The command `\newpageintoc` makes it possible to insert a page break inside the table of contents (toc). It may be used to avoid orphan titles in the toc.

5.24 Additional Document Sectioning Commands

The package `upmethodology-fmt` provides several commands that permit to create special sections.

5.24.1 Non-numbered Part in Table of Content

If you want to add a document part that has no part number but appearing inside the table of content, the classical LaTeX commands `\part` and `\part*` are inefficient. Indeed, `\part` is adding a numbered part inside the table of content, and `\part*` is adding an unnumbered part but not inside the table of content.

To add an unnumbered part inside the table of content, you could use one of the commands:

Syntax 1. `\parttoc[toctitle]{title}`

Syntax 2. `\parttoc*[toctitle]{title}`

The commands `\parttoc` and `\parttoc*` have the same effect except that `\parttoc*` aligns the part's title to the other numbered parts' titles; and `\parttoc` not.

5.24.2 Non-numbered Chapter in Table of Content

If you want to add a document chapter that has no chapter number but appearing inside the table of content, the classical LaTeX commands `\chapter` and `\chapter*` are inefficient. Indeed, `\chapter` is adding a numbered chapter inside the table of content, and `\chapter*` is adding an unnumbered chapter but not inside the table of content.

To add an unnumbered chapter inside the table of content, you could use one of the commands:

Syntax 1. `\chaptertoc[toctitle]{title}`

Syntax 2. `\chaptertoc*[toctitle]{title}`

The commands `\chaptertoc` and `\chaptertoc*` have the same effect except that `\chaptertoc*` aligns the chapter's title to the other numbered chapters' titles; and `\chaptertoc` not.

5.24.3 Non-numbered Section in Table of Content

If you want to add a document section that has no a section number but appearing inside the table of content, the classical LaTeX commands `\section` and `\section*` are inefficient. Indeed, `\section` add a numbered section inside the table of content, and `\section*` adds an unnumbered section but not inside the table of content.

To add an unnumbered section inside the table of content, you could use one of the commands:

Syntax 1. `\sectiontoc[toctitle]{title}`

Syntax 2. `\sectiontoc*[toctitle]{title}`

The commands `\sectiontoc` and `\sectiontoc*` have the same effect except that `\sectiontoc*` aligns the section's title to the other numbered sections' titles; and `\sectiontoc` not.

5.24.4 Non-numbered Subsection in Table of Content

If you want to add a document subsection that has no subsection number but appearing inside the table of content, the classical LaTeX commands `\subsection` and `\subsection*` are inefficient. Indeed, `\subsection` is adding a numbered subsection inside the table of content, and `\subsection*` is adding an unnumbered subsection but not inside the table of content.

To add an unnumbered subsection inside the table of content, you could use one of the commands:

Syntax 1. `\subsectiontoc[toctitle]{title}`

Syntax 2. `\subsectiontoc*[toctitle]{title}`

The commands `\subsectiontoc` and `\subsectiontoc*` have the same effect except that `\subsectiontoc*` aligns the subsection's title to the other numbered subsections' titles; and `\subsectiontoc` not.

5.24.5 Non-numbered Subsubsection in Table of Content

If you want to add a document subsubsection that has no subsubsection number but appearing inside the table of content, the classical LaTeX commands `\subsubsection` and `\subsubsection*` are inefficient. Indeed, `\subsubsection` is adding a numbered subsubsection inside the table of content, and `\subsubsection*` is adding an unnumbered subsubsection but not inside the table of content.

To add an unnumbered subsubsection inside the table of content, you could use one of the commands:

Syntax 1. `\subsubsectiontoc[toctitle]{title}`

Syntax 2. `\subsubsectiontoc*[toctitle]{title}`

The commands `\subsubsectiontoc` and `\subsubsectiontoc*` have the same effect except that `\subsubsectiontoc*` aligns the subsubsection's title to the other numbered subsubsections' titles; and `\subsubsectiontoc` not.

5.24.6 Chapter with different labels in TOC, headers and document

If you want to control the labels in the table of contents (TOC), the headers and the document for a chapter, the classical LaTeX commands `\chapter` and `\chapter*` are inefficient.

Syntax. `\chapterfull[toctitle]{title}{headertitle}`

The command create a chapter with the given label "`title`" in the core part of the document, with the given label "`toctitle`" in the table of contents, and with the label "`headertitle`" in the headers.

5.24.7 Section with different labels in TOC, headers and document

If you want to control the labels in the table of contents (TOC), the headers and the document for a section, the classical LaTeX commands `\section` and `\section*` are inefficient.

Syntax. `\sectionfull[toctitle]{title}{headertitle}`

The command create a section with the given label “`title`” in the core part of the document, with the given label “`toctitle`” in the table of contents, and with the label “`headertitle`” in the headers.

6

PACKAGE UPMETHODOLOGY-DOCUMENT

Version: 2025/04/20

The package `upmethodology-document` provides base functions to manage document information (project, subproject, authors...).

6.1 Document Information and Declaration

The informations associated to an UP document are:

- `\theupmproject` is the name of the project for which the document was produced;
- `\theupmsubproject` is the name of the sub-project for which the document was produced;
- `\theupmdocname` is the name of the document;
- `\theupmdocref` is the reference number of the document;
- `\theupmfulldocname` is the complete name of the document (composing by the project, subproject and name of the document).

You could declare the information about your document with one of the following functions:
`\declaredocument{project}{name}{ref}`
`\declaredocumentex{project}{subproject}{name}{ref}`
where the parameters are:

- `project` is the name of the project the document belongs to;
- `subproject` is the name of the sub-project the document belongs to;
- `name` is the name of the document;
- `ref` is the reference number of the document.

6.2 Abstract and Key-words

You are able to declare the abstract and the key-words for your document. Both are basically used by the back page package.

6.2.1 Declarations

The command `\setdocabstract` is for entering the docment's abstract:
`\setdocabstract[lang]{abstract_text}`
where `abstract_text` is the text of your abstract and `lang` designates for which language the abstract text is for. If the language is not specified, this command uses the current document language.

The command `\setdockeywords` is for entering the document's key-words:
`\setdockeywords[lang]{keywords}`
where `keywords` is the list of key-words and `lang` designates for which language the key-words are for. If the language is not specified, this command uses the current document language.

6.2.2 Rendering

The command `\theupmdocabstract` is expanded with the abstract text:
`\theupmdocabstract`

The command `\theupmdockeywords` is expanded with the key-words:
`\theupmdockeywords`

6.3 Document Summary

You can obtain a document summary with the command `\upmdocumentsummary[width]` which produces:

Document Summary	
Project	LaTeX Packages for Structured Documents as for Unified Process Methodology
Document	Official Documentation
Reference	UPM-2025-01
Version	34.0
Last Update	2025/04/29

6.4 Change Icons

By default, this package uses the logo of Arakhnê.org as icons. You could change them with the commands:

- `\defupmsmalllogo{filename}` defines the small logo used in the headers for instance;
- `\defupmlogo{filename}` defines the logo used on the front page for instance.

The logos' filenames are accessible with the functions `\theupmsmalldoclogo` and `\theupmdoclogo`.

6.5 Document Authors

An author is someone who participates to the writing of the document. You could register author identities with:

```
\addauthor[email]{firstname}{name}
\addauthor*[email]{firstname}{name}{comment}
\addauthorvalidator[email]{firstname}{name}
\addauthorvalidator*[email]{firstname}{name}{comment}
```

The list of the authors is accessible by two means:

- `\theauthorlist` is a coma-separated list of the authors' names;
- `\upmdocumentauthors` produces an array of all the authors (see below for an example).

Authors

Names	*Comments*	*Emails*
Stéphane Galland	Original Author	galland@arakhne.org
Frans van Dunné	Reviewer	

You could test if a string is the name of the author with:

- `\ifdocumentauthor{lowercasename}{then}{else}`; the first parameter **must** be lower case. If the `lowercasename` is the name of one of the authors, then the `then` clause is expanded, otherwise the `else` clause is expanded.

Authors

Names	*Comments*	*Emails*
Stéphane Galland	Original Author	galland@arakhne.org
Frans van Dunné	Reviewer	

6.6 Document Validators

A validator is someone who participates to the validation of the document. You could register validator identities with:

```
\addvalidator[email]{firstname}{name}
\addvalidator*[email]{firstname}{name}{comment}
\addauthorvalidator[email]{firstname}{name}
\addauthorvalidator*[email]{firstname}{name}{comment}
```

The list of the validators is accessible by two means:

- `\thevalidatorlist` is a coma-separated list of the validator's names;
- `\upmdocumentvalidators` produces an array of all the validators (see below for an example).

Validators			
Names	*Comments*	*Emails*	*Initials*
STÉPHANE GALLAND	Original Author	galland@arakhne.org	

6.7 Informed People

An informed people is someone who receives the document to be informed about its content. You could register informed people identities with:
`\addinformed[email]{firstname}{name}`
`\addinformed*[email]{firstname}{name}{comment}`

The list of the informed people is accessible by two means:

- `\theinformedlist` is a coma-separated list of the informed people's names;
- `\upmdocumentinformedpeople` produces an array of all the informed people (see below for an example).

6.8 Copyright and Publication Information

Package `upmethodology-document` provides several commands to define the copyright owner and the publication informations required to generate a publication page.

6.8.1 Setting Information

The Copyright holder(s) are person(s) or institution(s), that own the copyright on the document. The following command allows you to set the identity of the copyright holder in all parts of the documents:
`\setcopyrighter{name}`

Publisher is the people or the institution, or both, which is publishing the document. Basically it is the same the copyrighter (see above):
`\setpublisher{name}`

Some times, copyright laws depend on the location where the document is printed. The following command allows you to put a message in the publication page which is indicating where the document is printed:
`\setprintingaddress{address}`

Publications may be identifier by international identifiers. Package `upmethodology-document` supports ISBN, ISSN and DOI: `\setisbn{number}`
`\setissn{number}`
`\setdoi{number}`

The specific text may be provided for explaining the purpose of the document. The text is shown into the copyright page. In order to change the document's purpose, the following command is provided:

`\setdocumentpurpose{text}`

6.8.2 Retreiving Information

The information set by the commands described in the previous section may be retreived with the following commands:
`\theupmcopyrighter`
`\theupmpublisher`
`\theupmprintedin`
`\theupmisbn`
`\theupmissn`
`\theupmdoi`

6.8.3 Publication Page

The package `upmethodology-document` provides the `\upmpublicationpage` command which is displaying a empty page with publication informations and optionally set the page number (default value is −1). Figure 6.1 illustrates the publication page of this document.

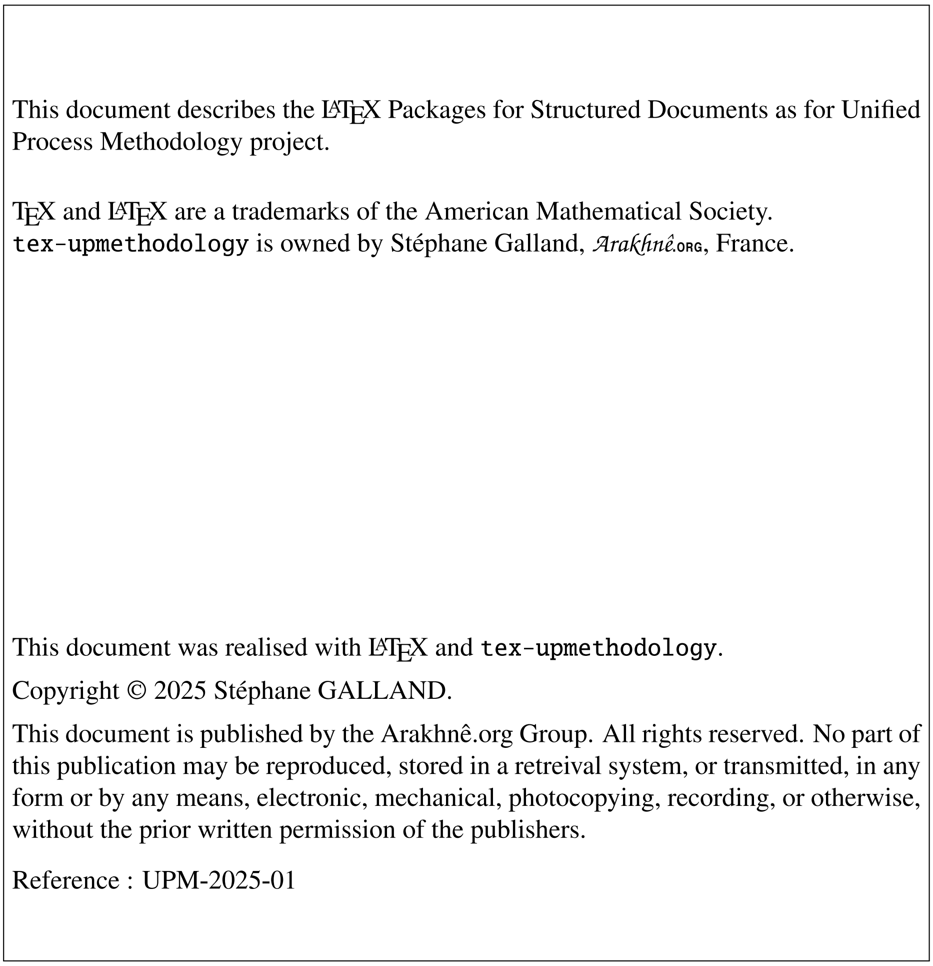
6.9 Localization

The current language is defined in the command `\upmcurrentlang`.

For testing the current language, you could use the command `\ifuplang{lang_id}{then commands}{else commands}`. This command tests if the given `lang_id` corresponds to the value expended by the command `\upmcurrentlang`. If it is true, the commands specified in the “then commands” are expanded. Otherwise, the commands specified in the “else commands” are expanded.

The following commands defines some localized strings used by `upmethodology-document`:

- `\upm@lang@project`: Project;
- `\upm@lang@document`: Document;
- `\upm@lang@docref`: Reference;
- `\upm@lang@lastupdate`: Last Update;
- `\upm@lang@document@summary`: Document Summary;
- `\upm@lang@document@authors`: Authors;
- `\upm@lang@document@validators`: Validators;
- `\upm@lang@document@names`: Names;
- `\upm@lang@document@emails`: Emails;
- `\upm@lang@document@initials`: Initials;
- `\upm@lang@document@abstract`: Abstract;
- `\upm@lang@document@keywords`: Key-words.



This document describes the LaTeX Packages for Structured Documents as for Unified Process Methodology project.

TeX and LaTeX are a trademarks of the American Mathematical Society.
`tex-upmethodology` is owned by Stéphane Galland, Arakhnê.org, France.

This document was realised with LaTeX and `tex-upmethodology`.

Copyright © 2025 Stéphane GALLAND.

This document is published by the Arakhnê.org Group. All rights reserved. No part of this publication may be reproduced, stored in a retreival system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publishers.

Reference : UPM-2025-01

Figure 6.1: Example of Publication Page generated with `\upmpublicationpage`

7

PACKAGE UPMETHODOLOGY-FRONTPAGE

`Version: 2025/04/20`

The `upmethodology-frontpage` package provides a front page for the UP documents. This package does not provides any public function. It is based on all the previous packages.

7.1 Display the front page

The front cover is displayed by invoking one of the following commands:

```
\maketitle
\makefrontcover
```

7.2 Change Front Page Layout

It is possible to change the layout of the front page with the command:

```
\setfrontlayout{layout_name}
```

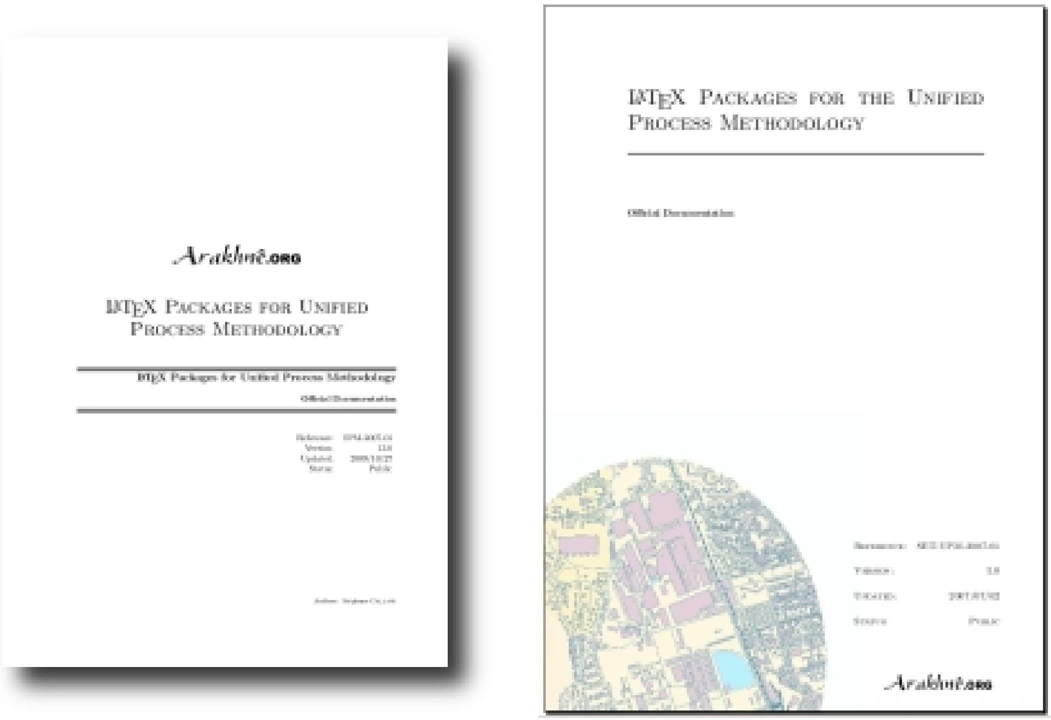
where `layout_name` must be one of:

- `classic`: classic front page layout with title and logo;
- `modern`: front page layout with title and logo and background picture.

The figure 7.1 illustrates the different layouts.

7.3 Change Illustration Picture

It is possible to insert an illustration picture on the front page. You could specify the image with the command:



LaTeX PACKAGES FOR UNIFIED PROCESS METHODOLOGY

LaTeX PACKAGES FOR THE UNIFIED PROCESS METHODOLOGY

(a) `classic` (b) `modern`

Figure 7.1: Front Page Layouts

`\setfrontillustration[width_factor]{filename}`
where:

- `width_factor` is the scaling factor of the picture according to the line width. If you specifies 1 the image will not be scaled, for `.5` the image will be the half of its original width...

- `filename` is the name of picture to use as the illustration.

7.4 Define a Front Page in Extensions

The `upmethodology-frontpage` package is able to use a page layout defined in a document extension (see chapter 9 for details on document extension).

A LaTeX command must be defined in the `upmext-NAME.cfg` file of the extension. The name of this command (for example `mylayout`) must be set with the `\set` command in the same file:

- `\Set{frontpage}{mylayout}` - for defining the entire content of the front page.
- `\Set{cfrontpage}{mycontent}` - for defining the content of the front page using the white page layout.
- `\Set{cfrontpage2}{mycontent}` - for defining the content of the second front page using the white page layout. The second page layout is rendered after the first front page.

7.5 Localization

The following commands defines some localized strings used by `upmethodology-frontpage`:

- `\upm@lang@front@authors`: Authors;

8

PACKAGE UPMETHODOLOGY-BACKPAGE

`Version: 2025/04/12`

The package `upmethodology-backpage` provides a back page for the UP documents. This package does not provides any public function. It is based on all the previous packages.

8.1 Display the back page

The back cover is displayed by invoking the following command:
`\makebackcover`

8.2 Change Back Page Layout

It is possible to change the layout of the back page with the command:
`\setbackcover{layout_name}`
where `layout_name` must be one of:

- `none`: no back page.

8.3 Small text before the back page

It is possible to insert a text at the bottom of the page just before the back page (usually the inner page of the cover for a two sided document). You must set the command `backcovermessage` with the `\Set` command:
`\Set{backcovermessage}{text}`

8.4 Define a Back Page in Extensions

The `upmethodology-backpage` package is able to use a page layout defined in a document extension (see chapter 9 for details on document extension).

A LaTeX command must be defined in the `upmext-NAME.cfg` file of the extension. The name of this command is `backpage`, and it must be set with the `\Set` command in the same file:
`\Set{backpage}{TeX commands}`

9

PACKAGE UPMETHODOLOGY-EXTENSION

`Version: 2025/04/20`

The package `upmethodology-extension` provides tools to create layout and rendering extensions. It is possible to write an extension to the `upmethodology-document` package. An extension is able to override several values from the default `upmethodology`-packages or may be used by the other suite's packages. For example, the Systems and Transport laboratory[1](15) extension is providing laboratory's icons, publisher's name and page layouts.

9.1 Load a Document Extension

To load and use a document extension, you must invoke the command:
`\UseExtension{extension_name}`
where `extension_name` is the identifier of the extension to load. The extension's files must be inside your LaTeX search path.

9.2 Write a Document Extension

A document extension could be written and described inside a file named `upmext-NAME.cfg`, where `NAME` is the name of the extension. This file must be put in your LaTeX search path.

The `upmext-NAME.cfg` file is a LaTeX file in which a set of definition commands are put. These commands must respect the LaTeX syntax.

The `\DeclareCopyright` command enables you to declare additional copyright information about the extension:
`\DeclareCopyright[lang]{extension_name}{year}{copyrighter}{trademark and copyright information}`

This command declares the `copyright` value which contains the copyright text (for this documentation “”). This command also declares the `trademarks` value which contains the trademark and other related informations about the extension (for this documentation “”).

Additional commands are provided to redefine the `upmethodology-document` constants:
`\Set[lang]{variable_name}{value}`

The `variable_name` is the name of the value to override. It must be taken in one of the names listed in table 9.1. The `lang` parameter is a language identifier. It is used to restrict the definition to a specific language. If not given, the default language is used instead. If it is given, the value is set only of the current language corresponds to the given `lang`. The `image_name` and `image_scale` are the name of the image file and the scaling factor respectively.

Value Name	*Description*
logo	the filename of the picture which must be used as a large logo.
smalllogo	the filename of the picture which must be used as a small logo.
copyrighter	the name of the authors or the institution which own the copyright on the document.
publisher	the name of the document's publisher. The `lang` parameter is supported.
printedin	the location/address where this document is printed.
frontillustration	the image to use as illustration. The `lang` parameter is ignored.
frontpage	the name of the front page style — not the LaTeX commands — to layout the front page. OR the front page illustration.
backpage	the LaTeX commands to layout the back page. OR the back page illustration.
cfrontpage	the LaTeX commands — not the name of the front page style — to layout the front page.

Table 9.1: List of overiddable value names

The `\SetLangDefault` command allows you to define a value for a specific language. It also change the language-independent value for the given variable name:
`\SetLangDefault{language}{variable_name}{text}`

The `\Get` command allows you to retrieve the value defined by a `\Set` for the current language:
`\Get{variable_name}`

The `\GetLang` command allows you to retrieve the value defined by a `\Set` or `\SetLangDefault` for a specific language:
`\GetLang{language}{variable_name}`

The `\Append` command allows you to append text to an existing definition of a value:
`\Append{variable_name}{text to append}`

The `\Unset` command allows you to remove the definition of a value:
`\Unset{variable_name}`

The `\Ifnotempty` command allows you to expand the LaTeX commands if the given text is not empty:

`\Ifnotempty{text}{latex_code}`

The `\Ifempty` command allows you to expand the LaTeX commands if the given text is empty:
`\Ifempty{text}{latex_code}`

The `\Ifelsedefined` command allows you to expand the LaTeX commands in `then_code` if a value with the given name was defined, or to expand the LaTeX commands in `else_code` if no value with the given name was defined:
`\Ifelsedefined{value_name}{then_code}{else_code}`

The `\Put` command is an extension of the standard picture `\put` command. It takes into account the joint margin applied in two sided documents when it is used on page's backside (eg. the back page of the document):
`\Put(x,y){commands}`

This command must be used inside a `picture` environment in place of the standard `\put` command.

10

PACKAGE UPMETHODOLOGY-TASK

`Version: 2025/04/25`

The LaTeX package `upmethodology-task` provides a set of commands to define project's tasks.

During LaTeX compilation this package could log the message `"Project Task(s) may have changed. Rerun to get cross-references right"` when some task information was not found or due to cross-references on them.

10.1 Task Definition

The definition of a task could be made only inside one of the following environments:
`\begin{taskdescription}{id}...\end{taskdescription}`
`\begin{taskdescription*}{id}...\end{taskdescription*}`
where `id` is the identifier of the task.

The environment `taskdefinion` displays the task's description with a call to `\thetaskdescription{id}`. On the other hand, `taskdefinition*` never displays the task's description.

Inside one of the task's definition environment above, you could use one of the following commands to define the task's attributes:

- `\taskname{name}`
  to define the name of the task;
- `\tasksuper{id}`
  indicates that the current task is a sub-task of the task identified by the given identifier;
- `\taskcomment{text}`
  to describe the task's purposes and goals (will be shown in the description box of the task's description);
- `\taskprogress{percent}`
  to set the percentage for task achievement;
- `\taskstart{date}`
  to set the starting date of the task (real or predicted);
- `\taskend{date}`
  to set the finished date of the task (real or predicted);

- `\taskmanager{name}`
  to add a task's manager into the list of the managers;
- `\taskmember{name}`
  to add a task's member into the list of the members;
- `\taskmilestone{date}{comment}`
  to add a milestone into the task for the given date and described by the given comment.

10.2 Task Reference

You could reference any information about the defined tasks in your document. In case you used cross-references this package could log the message "`Project Task(s) may have changed. Rerun to get cross-references right`" to complain about rebuilding of our document.

The following commands are available:

- `\thetasksuper{id}`
  replies the identifier of the parent task corresponding to the task identified by `id`;
- `\thetaskname{id}`
  replies the name of the the task identified by `id`;
- `\thetaskcomment{id}`
  replies the description for the the task identified by `id`;
- `\thetaskprogress{id}`
  replies the archieving percent for the the task identified by `id`;
- `\thetaskstart{id}`
  replies the starting date for the the task identified by `id`;
- `\thetaskend{id}`
  replies the ending date for the the task identified by `id`;
- `\thetaskmanagers{id}`
  replies the managers' list for the the task identified by `id`;
- `\thetaskmembers{id}`
  replies the members' list for the the task identified by `id`;
- `\thetaskmilestones{id}`
  replies the list of milestone's dates for the the task identified by `id`;
- `\thetaskmilestonecomment{id}{date}`
  replies the comment of the given milestone for the the task identified by `id`;
- `\thetaskdescription[width]{id}`
  replies the complete description of the the task identified by `id`.

10.3 Localization

The following commands defines some localized strings used by `upmethodology-task`:

- `\upm@task@lang@task`: Task;

- `\upm@task@lang@escription`: Description;
- `\upm@task@lang@startat`: Start at;
- `\upm@task@lang@endat`: End at;
- `\upm@task@lang@archieved`: Achieved;
- `\upm@task@lang@managers`: Managers;
- `\upm@task@lang@members`: Members;
- `\upm@task@lang@Milestones`: Milestones;
- `\upm@task@lang@subtask`: Sub-task of.

10.4 Example of task

Task *id:tache*: nom		
Description: Sub-task of:	description Task *id:tache:0*	
Start at: 2025/04/24	End at: 2026/04/24	Archieved: percentage%
Managers: Members:	Stephane Galland Stephane Galland	
Milestones:	• **2025/07/24:** Description	

11

PACKAGE UPMETHODOLOGY-CODE

`Version: 2025/04/12`

The LaTeX package `upmethodology-code` provides a set of commands for source code formatting. The supported source codes are UML, Java and C++.

You could load the package with the following options:

`uml`	use the UML notation (default value)
`java`	use the Java notation
`cpp`	use the C++ notation

You could also change the notation language with the command:
`\upmcodelang{upm|java|cpp}`

The provided commands are listed in the following table:

command	UML	Java	C++
Prototypes			
`\jclass{TheClass}`	TheClass	TheClass	TheClass
`\jinterface{TheInterface}`	*TheInterface*	*TheInterface*	*TheInterface*
`\jpackage{ThePackage}`	ThePackage	ThePackage	ThePackage
`\jfunc{FunctionName}`	`FunctionName`	`FunctionName`	`FunctionName`
Types			
`\jclazz`	**class**	**Class**	**class**
`\jvoid`	**void**	**void**	**void**
`\jboolean`	**boolean**	**boolean**	**bool**
`\jint`	**integer**	**int**	**int**
`\jlong`	**long integer**	**long**	**long**
`\jfloat`	**float**	**float**	**float**
`\jdouble`	**double**	**double**	**double**
`\jchar`	**character**	**char**	**char**
`\jstring`	**string**	String	std::string
`\jarray{T}`	**array of** Ts	T[]	T[]
`\jcollection{T}`	**collection of** Ts	Collection <T>	std::vector <T>
`\jset{T}`	**set of** Ts	Set <T>	std::set <T>

command	UML	Java	C++
Constants			
`\jtrue`	TRUE	TRUE	TRUE
`\jfalse`	FALSE	FALSE	FALSE
Operations			
`\jcode{source code}`	`source code`	`source code`	`source code`
`\jcall{fct}{params}`	`fct(params)`	`fct(params)`	`fct(params)`
`\jop{operator}`	`operator`	`operator`	`operator`

12

AUTHORS AND LICENSE