autolatex - compile LaTeX and TeX documents

NAME

autolatex - compile LaTeX and TeX documents

SYNOPSIS

```
autolatex [global options] command [specific options]
```

DESCRIPTION

AutoLaTeX is a tool for managing small to large LaTeX documents. It simplifies the process of producing PDF files or viewing documents by tracking file changes and automating the execution of required programs. One of AutoLaTeX's best features is its translator rules (also known as translators), which automatically generate drawings for inclusion in the PDF from various source formats such as SVG and GNU Plot.

For example, consider a document with a single LaTeX file, `mydoc.tex`. Without AutoLaTeX, producing a `.pdf` file might require the following commands:

```
pdflatex mydoc.tex
bibtex mydoc
makeindex mydoc
makeglossaries mydoc
pdflatex mydoc.tex
pdflatex mydoc.tex
pdflatex mydoc.tex
```

The triple invocation of LaTeX ensures that all references are resolved and layout changes are accounted for. While not overly complex, this process still involves multiple commands. With AutoLaTeX, you can simplify this to:

```
autolatex -f mydoc.tex
```

or simply:

```
autolatex
```

For documents requiring additional tools to generate PDF versions of included drawings or to run BibTeX/Biber for bibliographies, manual compilation becomes cumbersome. AutoLaTeX automates these tasks: it calls the necessary translators and tools for you. Each translator converts a source file (e.g., EPS, SVG, Gnuplot) into a PDF or PNG file.

This introduction demonstrates how AutoLaTeX simplifies LaTeX document management. The system is user-friendly for small projects and robust enough for large ones. The rest of this manual provides comprehensive documentation on using, configuring, and installing AutoLaTeX.

COMMANDS

AutoLaTeX provides a fixed set of commands to perform specific actions on your document. The default command is **build**. Some commands accept optional arguments; use `--help` after the command name for details.

The available commands are:

biblio

Processes all tasks required to generate the bibliography (BibTeX, Biber, etc.).

build

Equivalent to the **images**, **document** and **view** commands, but the viewer is launched only if enabled in the configuration or via the command line.

clean

Removes all LaTeX temporary files and other temporary files created during project processing from the current working directory. The drawings that are automatically generated by the **images** command are not removed.

cleanall

Same as **clean**, but also removes Emacs ~ files, other backup files, generated figures, and the produced PDF. This command is also known as **mrproper**.

createbeamer

Creates a LaTeX package file in the document directory that equips LaTeX Beamer text processor with the control sequences dedicated to AutoLaTeX (slides with animated figures). The package file is named `autolatex-beamer.sty`. See the command **createsty** for creating a package dedicated to regular LaTeX document.

createconfig

Creates a standard configuration file in the current directory based on the command-line configuration.

createist

Creates a default MakeIndex style file in the document directory, named `default.ist`.

createsty

Creates a LaTeX package file in the document directory that equips LaTeX text processor with the control sequences dedicated to AutoLaTeX (animated figures and figures with embedded TeX). The package file is named `autolatex.sty`. See the command **createbeamer** for creating a package dedicated to LaTeX Beamer.

commit

Commits changes to a version control system (GIT, CVS, SVN…). The command line must be provided in the configuration file.

document

Performs all processing required to produce the `.pdf`, `.dvi`, or `.ps` file for the document. This command is also known as **gen_doc**.

glossaries

Performs all processing required to generate glossaries (makeglossaries). This command is also known as **makeglossaries**.

images

Automatically generates drawings by invoking the translators.

index

Performs all processing required to generate the index (makeindex). This command is also known as **makeindex**.

init

Creates an empty LaTeX document in the current directory following a standard folder structure supported by AutoLaTeX.

latex

Run one time the (La)TeX text processing to produce the `.pdf`, `.dvi`, or `.ps` file for the document. This command is also known as **tex** or **maketex**.

makeflat

Creates a version of the document in a specific folder (default: `./flat_version/`) with a single LaTeX or TeX file and all other files in the same directory, excluding subfolders. This command is also known as **preparepublish**.

This command is useful for creating a version of the document that can be directly uploaded to online publication sites (e.g., Elsevier), which do not support subfolder uploads.

This command provides the CLI option `--externalbiblio` to specify whether the bibliography should be placed in a BibTeX file (external) or inlined in the TeX file (default: inlined).

showbuildprocess

Displays the list of actions that will be applied by AutoLaTeX during the building process. This function does not start the building process.

showconfig

Displays the configuration definitions read from the configuration files.

showconfigfiles

Displays the list of configuration filenames read by AutoLaTeX.

showdependencies

Displays the dependencies of the main LaTeX document in terms of included and used

files.

showimages

Displays the filenames of figures that should be processed by a translator. This command offers specific options to control the type of information displayed.

showinstalledtranslators

Displays the list of installed translators. This command is also known as **installedtranslators**.

showloadedtranslators

Displays the list of loaded translators. This command is also known as **translators**.

showpath

Displays the value of the PATH environment variable that is considered by AutoLaTeX.

stamps

Displays or update the list of stamps that are saved by AutoLaTeX. Stamps are keys used to determine if some information (bibliography, index, glossary, etc.) has changed since the previous run of AutoLaTeX. They permit to AutoLaTeX to obtain a better detection of changes than those only based on the dates of last-change of the files.

unusedimages

Displays (or removes) figures in the document folder that are not included in the document.

update

Updates the local copy with changes from a version control system (GIT, CVS, SVN…). The command line must be provided in the configuration file.

view

Launches the document viewer. The command line of the viewer must be provided in the configuration file.

GLOBAL OPTIONS

`--asyncview`, `--noasyncview`

Enables or disables asynchronous viewer launching. If enabled, AutoLaTeX does not wait for the viewer to close before stopping execution. If disabled, AutoLaTeX waits for the viewer to close.

`--auto`, `--noauto`

Enables or disables automatic figure generation using translators.

`--biblio`, `--nobiblio`

Enables or disables the bibliography tool (BibTeX, Biber, etc.) if not explicitly invoked

from the command line.

`--continuous [sleep_duration], --nocontinuous`

Runs AutoLaTeX continuously, repeatedly performing the specified actions. This option causes AutoLaTeX to loop infinitely, similar to the following script (in bash):

```
while 1
do
  autolatex "$@"
  sleep sleep_duration
done
```

The `sleep_duration` parameter specifies the waiting time in seconds between loops. If not provided, it defaults to 0. The `--continuous` option forces `--asyncview` to be enabled.

With a compatible viewer, the display will update automatically. Some UNIX/Linux versions of "gv -watch" support this for PostScript files, which can be configured via a variable. Many other previewers require manual updates.

Note: Adobe Acrobat Reader on MS-Windows locks the PDF file, preventing updates, so it is not recommended for continuous mode.

`--debug`

Runs AutoLaTeX with the logging level set to 'debug'. The available levels (from least to most verbose) are: off, critical, error, warning, info, fine_info, debug, trace.

`--defaultist`

Allows AutoLaTeX to use MakeIndex with the default style (.ist file). The default style is provided by the AutoLaTeX distribution. The `--index` and `--noindex` options modify AutoLaTeX's behavior regarding MakeIndex.

`-d directory, --directory directory`

Specifies a directory containing a LaTeX document to compile. You can specify this option for each directory containing a LaTeX document. If not specified, the current directory is used.

`--dvi`

Specifies that the result of the AutoLaTeX process is a DVI or XDVI document, which will be converted to PostScript. This option has the same effect as `--ps`.

`-e name, --exclude name`

Prevents AutoLaTeX from loading the translator named `name`. See below for available translators. The `--include` option includes a translator, and `--include-path` specifies where to find translator scripts.

`--extramacro, --noextramacro`

Enables or disables the supports of extra TeX and LaTeX macros and environments. The list of these extra definitions in detailed in the dedicated section of this documentation.

`-f file, --file file`

Specifies the main LaTeX file to compile. If not specified, AutoLaTeX searches for a TeX file in the document directory.

`--file-line-warning`, `--nofile-line-warning`

Experimental: Enables or disables the extended warning format for LaTeX. This format adds the filename and line number to the warning message, which is useful for extracting warnings from the log file.

`--gloss`, `--glossary`, `--nogloss`, `--noglossary`

Enables or disables the use of MakeGlossaries.

`-h`, `--help`

Displays the manual. If used before any command, the global manual is shown. If used after a command, only the manual page for that command is displayed.

`-D directory`, `--imgdirectory directory`

Specifies a directory where AutoLaTeX will find figures to be processed by translators. Each use of this option adds a directory to the list.

`-i name`, `--include name`

Forces AutoLaTeX to load the translator named `name`. See below for available translators. The `--exclude` option excludes a translator, and `--include-path` specifies where to find translator scripts.

`-I paths`, `--include-path paths`

Notifies AutoLaTeX of directories containing translator scripts. `paths` can be a list separated by the operating system's path separator (':' on Unix, ';' on Windows). The `--exclude` option excludes a translator, and `--include` includes a translator.

`--index [style_file]`, `--noindex`

Allows AutoLaTeX to use MakeIndex. If a value is provided, it is assumed to be an .ist file for MakeIndex. If no value is provided, AutoLaTeX uses MakeIndex and attempts to detect a MakeIndex style file (.ist) in the document directory. If none is found, no style is passed to MakeIndex. The `--defaultist` and `--noindex` options modify AutoLaTeX's behavior regarding MakeIndex.

`--info`

Runs AutoLaTeX with the logging level set to 'info'. The available levels (from least to most verbose) are: off, critical, error, warning, info, fine_info, debug, trace.

`--latex`

Uses the LaTeX command: `latex`. See also: `--pdflatex`, `--lualatex`, and `--xelatex`.

`--lualatex`

Uses the LaTeX command: `lualatex`. See also: `--pdflatex`, `--latex`, and `--xelatex`.

`--pdf`

Specifies that the result of the AutoLaTeX process is a PDF document.

`--pdflatex`

Uses the LaTeX command: `pdflatex`. See also: `--latex`, `--lualatex`, and `--xelatex`.

`--ps`

Specifies that the result of the AutoLaTeX process is a PostScript document. This option has the same effect as `--dvi`.

`-q`, `--quiet`

Runs AutoLaTeX with only critical and error messages. The available levels (from least to most verbose) are: off, critical, error, warning, info, fine_info, debug, trace.

`--search-project-from file`

When specified, AutoLaTeX searches for a configuration file (usually `.autolatex_project.cfg` on Unix) in the directory of the specified `file` or its ancestors. This option does not replace the `-d` or `-f` options.

`--showloglevel`

Show the current level of logging on the console. This option depends on the usage of the other logging options, such as `--verbose` or `--debug` for example.

`--silent`

Runs AutoLaTeX without logging messages (off logging level). The available levels (from least to most verbose) are: off, critical, error, warning, info, fine_info, debug, trace.

`--stderr`, `--stdout`

Directs AutoLaTeX to output regular messages (not logged) to standard output (stdout) or standard error (stderr).

`--synctex`, `--nosynctex`

Enables or disables the generation of SyncTeX-compatible output files. SyncTeX links a viewer and a TeX editor, allowing you to click in one and highlight the corresponding line in the other.

`--testlogs`

Show the a message for each level of logging. This option enables to show the behavior of the logging system that is used by AutoLaTeX. This option depends on the usage of the other logging options, such as `--verbose` or `--debug` for example.

`-v`, `--verbose`

Increases verbosity each time this option is specified. The available levels (from least to most verbose) are: off, critical, error, warning, info, fine_info, debug, trace.
Note: If you specify more verbosity than ‘trace’, AutoLaTeX stops immediately and displays the current in-memory configuration.

`--version`

Displays the version of AutoLaTeX.

`--view`, `--noview`

Enables or disables the document viewer at the end of compilation.

`--Wall`

Runs AutoLaTeX with the logging level set to ‘fine_info’. The available levels (from least to most verbose) are: off, critical, error, warning, info, fine_info, debug, trace.

`--Wnone`

Runs AutoLaTeX with the logging level set to ‘error’, suppressing warnings. The available levels (from least to most verbose) are: off, critical, error, warning, info, fine_info, debug, trace.

`--xelatex`

Uses the LaTeX command: `xelatex`. See also: `--pdflatex`, `--latex`, and `--lualatex`.

OPTIONS FOR COMMANDS

`biblio`

`--nochdir`

Prevents AutoLaTeX from changing the current directory to the document’s root directory before starting the build process.

`build`

See options for the `images`, `document` and `view` commands, if any.

`clean`

`--nochdir`

Prevents AutoLaTeX from changing the current directory to the document’s root directory before starting the cleaning process.

`--norecursive`

Disables cleaning in subfolders.

`--simulate`

Simulates file removal without actually deleting files.

`--all`

If specified, the clean command behaves like the `cleanall` command.

`cleanall`

See the options for `clean`, except for `--all`, which is implicit for `cleanall`.

createbeamer

`--force`

Overwrites the .sty file for LaTeX Beamer if it exists.

createconfig

`--force`

Overwrites the configuration file if it exists.

createist

`--force`

Overwrites the .ist file if it exists.

createsty

`--force`

Overwrites the .sty file for LaTeX if it exists.

document

`--force`

Force the building of all the intermediary files (bbl, idx, ind, gls, etc.) even if they are up-to-date.

`--nochdir`

Prevents AutoLaTeX from changing the current directory to the document's root directory before starting the build process.

glossaries

`--nochdir`

Prevents AutoLaTeX from changing the current directory to the document's root directory before starting the build process.

images

`--force`

Forces the overwriting of all generated figures, ensuring AutoLaTeX runs the translators for all of them.

index

`--nochdir`

Prevents AutoLaTeX from changing the current directory to the document's root

directory before starting the build process.

init

`--force`

Overwrites all generated files and folders if they exist.

`--out directory`

Specifies the directory where the generated document structure will be written.

latex

`--nochdir`

Prevents AutoLaTeX from changing the current directory to the document's root directory before starting the build process.

`--showneedloop`

Show if another run of the (La)TeX text processing is detected as needed for producing the .pdf, .dvi, .ps file.

makeflat

`--externalbiblio`

Forces the use of an external BibTeX file (i.e., a .bib file) instead of inlining the bibliography database in the TeX file.

`--out directory`

Specifies the output directory for the flat version. The default directory name is `flat_version`.

showdependencies

`--list`

Display the dependencies in the form of list of filenames in place of the default tree of dependencies.

`--noauxfile`

Prevents AutoLaTeX from reading the auxilliary files in order to extract file dependencies.

`--times`

Display for each file the last-change time that is considered by AutoLaTeX in the processing stages.

showimages

`--changed`

Shows only images not associated with up-to-date generated files, i.e., images requiring translator processing.

`--translators`

Shows all images and their associated translators.

`--valid`

Shows only images associated with up-to-date generated files, i.e., images that do not require translator processing.

`showinstalledtranslators`

`--level`, `--nolevel`

Shows or hides the installation levels for each translator.

`showloadedtranslators`

`--level`, `--nolevel`

Shows or hides the activation levels for each translator.

`stamps`

`--reset`

Reset to nothing the values of the stamps.

`--update`

Force the compuation of the stamps from the current content of the axuiliary files, and save the stamps.

`unusedimages`

`--delete`

Deletes unused images instead of just listing them.

AUTO GENERATION OF FIGURES

A converter, called a translator, transforms a source figure into a target format supported by LaTeX. This converter can be an external program (e.g., epstopdf) or an internal Python, Perl, Ruby, or Bash script.

Each supported translator is defined in a `.transdef2` file (the old format in the Perl implementation of AutoLaTeX uses `.transdef`). This file contains the shell command line or code to execute. To create a new translator, copy and modify an existing `.transdef2` file. Even if excluded from the command line, a translator is automatically included by AutoLaTeX if invoked by another included translator.

The provided translators are (only pdf outputs are considered below, ps outputs are also

possible):

- **Astah/Jude (asta) to Portable Document Format (pdf)**
  - Name: `astah2pdf`
  - Use external converter: `astah-pro`, `astah-uml`, `astah-com`
  - Use translator: `svg2pdf`
  - Input format: `.asta .jude .juth`
  - Output format: `.pdf`
- **Astah/Jude (asta) to Portable Network Graphic (png)**
  - Name: `astah2png`
  - Use external converter: `astah-pro`, `astah-uml`, `astah-com`
  - Input format: `.asta .jude .juth`
  - Output format: `.png`
- **Astah SysML (asml) to Portable Document Format (pdf)**
  - Name: `asml2pdf`
  - Use external converter: `astah-sysml`
  - Use translator: `svg2pdf`
  - Input format: `.asml`
  - Output format: `.pdf`
- **Asymptote (asy) to Portable Document Format (pdf)**
  - Name: `asy2pdf`
  - Use external converter: `asy`
  - Use translator: `eps2pdf`
  - Input format: `.asy`
  - Output format: `.pdf`
- **C/C++ Source Code (.cpp, .c, .hpp, .h) to TeX Source Code (tex): TeXify variant**
  - Name: `cpp2tex_texify`
  - Use external converter: `texifyc++`

- Input format: `.cpp, .c, .hpp, .h, .c++, .h++`
- Output format: `.tex`

- **Compressed Bitmap to Uncompressed Bitmap**
  Based on `zcat`. This translator assumes input files are compressed with the '.gz' extension. It allows storing compressed figures in the project as raw material for the LaTeX compiler. The bitmaps are decompressed into a file with the same name, excluding the '.gz' extension.
  - Name: `imggz2img`
  - Use external converter: `zcat`
  - Input format: `XXX.gz`
  - Output format: `XXX`
- **Diagram Editor (dia) to Portable Document Format (pdf)**
  - Name: `dia2pdf`
  - Use external converter: `dia`
  - Use translator: `eps2pdf`
  - Input format: `.dia`
  - Output format: `.pdf`
- **Diagram Editor (dia) to TeX embedded in Portable Document Format (pdf+tex)**
  - Name: `dia2pdf+tex`
  - Use external converter: `dia`
  - Input format: `.dia_tex .diat .dia+tex .diatex .tex.dia +tex.dia`
  - Output formats: `.pdf`, `.pdftex_t`
- **Dot Graphviz (dot) to Portable Document Format (pdf)**
  - Name: `dot2pdf`
  - Use external converter: `dot`
  - Input format: `.dot`
  - Output format: `.pdf`
- **Dot Graphviz (dot) to Portable Network Graphic (png)**
  - Name: `dot2png`

    - Use external converter: `dot`
    - Input format: `.dot`
    - Output format: `.png`
- **Dot Graphviz (dot) to TeX (tex)**
    - Name: `dot2tex`
    - Use external converter: `dot`
    - Input format: `.dot`
    - Output format: `.tex`
- **Encapsulated PostScript (eps) to Portable Document Format (pdf)**
    - Name: `eps2pdf_epstopdf`
    - Use external converter: `epstopdf`
    - Input format: `.eps`
    - Output format: `.pdf`
- **XFig document (fig) to Portable Document Format (pdf)**
    - Name: `fig2pdf`
    - Use external converter: `fig2dev`
    - Input format: `.fig`
    - Output format: `.pdf`
- **XFig document (fig) to TeX embedded in Portable Document Format (pdf+tex)**
    - Name: `fig2pdf+tex`
    - Use external converter: `fig2dev`
    - Input format: `.figt .fig_tex .figtex .fig+tex .tex.fig +tex.fig`
    - Output formats: `.pdf`, `.pdftex_t`
- **Graph eXchange Language (gxl) to Portable Document Format (pdf)**
    - Name: `gxl2pdf`
    - Use external converter: `gxl2dot`
    - Use translator: `dot2pdf`

    - Input format: `.gxl`
    - Output format: `.pdf`
- **Graph eXchange Language (gxl) to Portable Network Graphic (png)**
    - Name: `gxl2png`
    - Use external converter: `gxl2dot`
    - Use translator: `dot2png`
    - Input format: `.gxl`
    - Output format: `.png`
- **Graphics Layout Engine (gle) to Portable Document Format (pdf)**
    - Name: `gle2pdf`
    - Use external converter: `gle`
    - Input format: `.gle`
    - Output format: `.pdf`
- **Graphics Layout Engine (gle) to Portable Network Graphic (png)**
    - Name: `gle2png`
    - Use external converter: `gle`
    - Input format: `.gle`
    - Output format: `.png`
- **Java Source Code (java) to TeX Source Code (tex): TeXify variant**
    - Name: `java2tex_texify`
    - Use external converter: `texifyjava`
    - Input format: `.java`
    - Output format: `.tex`
- **Lisp Script (lisp) to TeX Source Code (tex): TeXify variant**
    - Name: `lisp2tex_texify`
    - Use external converter: `texifyLisp`
    - Input format: `.lisp`

    - Output format: `.tex`
- **MatLab Script (m) to TeX Source Code (tex): TeXify variant**
    - Name: `matlab2tex_texify`
    - Use external converter: `texifymatlab`
    - Input format: `.m`
    - Output format: `.tex`
- **ML Script (ml) to TeX Source Code (tex): TeXify variant**
    - Name: `ml2tex_texify`
    - Use external converter: `texifyml`
    - Input format: `.ml`
    - Output format: `.tex`
- **Perl Script (perl) to TeX Source Code (tex): TeXify variant**
    - Name: `perl2tex_texify`
    - Use external converter: `texifyperl`
    - Input format: `.perl .pl`
    - Output format: `.tex`
- **Gnuplot (plot) to Portable Document Format (pdf)**
    - Name: `plot2pdf`
    - Use external converter: `gnuplot`
    - Use translator: `eps2pdf`
    - Input format: `.plot .gnu`
    - Output format: `.pdf`
- **Gnuplot (plot) to TeX embedded in Portable Document Format (pdf+tex)**
    - Name: `plot2pdf+tex`
    - Use external converter: `gnuplot`
    - Use translator: `eps2pdf`
    - Input format: `.plott .plot_tex .plottex .plot+tex .tex.plot`

`+tex.plot .gnut .gnu_tex .gnutex .gnu+tex .tex.gnu +tex.gnu`

- Output formats: `.pdf`, `.pdftex_t`

- **Python Source Code (py) to TeX Source Code (tex): TeXify variant**
  - Name: `python2tex_texify`
  - Use external converter: `texifyPython`
  - Input format: `.py`
  - Output format: `.tex`
- **Ruby Source Code (rb) to TeX Source Code (tex): TeXify variant**
  - Name: `ruby2tex_texify`
  - Use external converter: `texifyRuby`
  - Input format: `.rb`
  - Output format: `.tex`
- **SQL Script (sql) to TeX Source Code (tex): TeXify variant**
  - Name: `sql2tex_texify`
  - Use external converter: `texifySQL`
  - Input format: `.sql`
  - Output format: `.tex`
- **Scalable Vector Graphic (svg) to Portable Document Format (pdf): Inkscape variant**
  - Name: `svg2pdf_inkscape`
  - Use external converter: `inkscape`
  - Input format: `.svg`
  - Output format: `.pdf`
- **Scalable Vector Graphic (svg) to TeX embedded in Portable Document Format (pdf+tex)**
  - Name: `svg2pdf+tex_inkscape`
  - Use external converter: `inkscape`
  - Input format: `.svgt .svg_t .svgtex .svg+tex .tex.svg +tex.svg`
  - Output formats: `.pdf`, `.pdftex_t`

- **Scalable Vector Graphic (svg) to Portable Network Graphic (png): Inkscape variant**
  - Name: `svg2png_inkscape`
  - Use external converter: `inkscape`
  - Input format: `.svg`
  - Output format: `.png`
- **Scalable Vector Graphic with layers (svg) to Beamer Overlays**
  - Name: `svg2pdf+layers_inkscape`
  - Use external converter: `inkscape`
  - Input format: `.svgl .svg_l .svglayers .svg+layers .layers.svg +layers.svg`
  - Output formats: `.pdf`, `.pdftex_t`
- **Scalable Vector Graphic with layers (svg) to TeX embedded in Beamer Overlays**
  - Name: `svg2pdf+layers+tex_inkscape`
  - Use external converter: `inkscape`
  - Input format: `.svglt .svg_lt .svglayerstex .svgtexlayers .svg+layers+tex .svg+tex+layers .layers.tex.svg .tex.layers.svg +layers+tex.svg +tex+layers.svg`
  - Output formats: `.pdf`, `.pdftex_t`
- **PGF/TikZ (tikz) to Portable Document Format (pdf)**
  - Name: `tikz2pdf`
  - Use external converter: `pdflatex`
  - Input format: `.tikz`
  - Output format: `.pdf`
- **Gimp (xcf) to Portable Document Format (pdf)**
  - Name: `xcf2pdf`
  - Use external converter: `convert`
  - Input format: `.xcf`
  - Output format: `.pdf`
- **Gimp (xcf) to Portable Network Graphic (png)**

    - Name: `xcf2png`
    - Use external converter: `convert`
    - Input format: `.xcf`
    - Output format: `.png`
- **UML Metadata Interchange (xmi) to Portable Document Format (pdf): Umbrello variant**
    - Name: `xmi2pdf_umbrello`
    - Use external converter: `umbrello`
    - Use translator: `eps2pdf`
    - Input format: `.xmi`
    - Output format: `.pdf`
- **Visio Binary Draw (vsd) to Portable Document Format (pdf)**
    - Name: `vsd2pdf`
    - Use external converter: `inkscape`
    - Input format: `.vsd .vdx .vsdx`
    - Output format: `.pdf`

ANIMATED FIGURES IN AUTOLATEX

Animated figures represent a powerful mechanism for constructing dynamic visual content within LaTeX Beamer presentations. Unlike conventional static graphics, an animated figure consists of multiple visual layers that can appear progressively across the slides of a single frame. This approach enables the presenter to control the incremental disclosure of complex diagrams, illustrations, or schematics, thereby guiding the audience through a narrative sequence without requiring separate image files for each step.

AutoLaTeX automates the generation of such figures by leveraging layered Scalable Vector Graphics (SVG) files and translating them into a Beamer-compatible format. The underlying principle relies on the mapping between layers in an SVG document and overlay specifications in Beamer. Each layer within the SVG file corresponds to a distinct visual element that may be shown or hidden on specific slides. To control this behavior, the author embeds a frame specification directly into the layer's name. This specification follows the format `<spec>`, where `spec` defines a set of slide numbers. The syntax accommodates single numbers (e.g., `<2>`), inclusive ranges (e.g., `<3-6>`), open-ended ranges (e.g., `<5->` for all slides from five onward, or

`<-8>` for slides up to and including eight), and combinations thereof (separated by coma characters). When the SVG is processed by the appropriate AutoLaTeX translator (such as `svg2pdf+layers_inkscape`) these specifications are extracted and used to generate a Beamer overlay sequence. The resulting output comprises a set of PDF files, each corresponding to one layer, along with a TeX file that orchestrates their inclusion using Beamer's overlay mechanisms. Consequently, when the presentation is compiled, the figure animates: each layer appears exactly on the slides designated by its embedded specification, and layers without a specification typically appear from the beginning.

Integration with Beamer is achieved through the LaTeX macros provided by the `autolatex-beamer.sty` package (see below).

Creating an animated figure suitable for AutoLaTeX and Beamer begins with designing the vector graphic in a tool that supports layers, such as Inkscape. The author constructs the diagram as a set of layers, where each layer represents one logical step of the animation. For instance, a technical drawing might have a base layer containing the background and axes, followed by successive layers adding curves, labels, or annotations. The crucial step lies in naming each layer to encode its appearance schedule. The author opens the layer properties dialog and sets the layer name to include the desired frame specification, such as "Base layer" for a layer that should always appear, or "Data curve <2-4>" for a layer that should appear only on slides two through four. After naming, the author saves the file with an extension that triggers the appropriate translator; typically, the `+layers.svg` extension is used to indicate that the file contains layered content intended for animation. AutoLaTeX's translator will then parse the layer names, generate the overlay structure, and produce the final assets ready for inclusion in the Beamer document.

This workflow combines the expressive power of vector graphics editing with the precise overlay control of Beamer, all automated by AutoLaTeX. It eliminates the manual effort of exporting multiple static images and coding their appearance order, thereby streamlining the creation of sophisticated, animated presentations.

FIGURES WITH EMBEDDED TEX CODE

In scientific and technical documents, figures often require textual elements that demand the typographic quality and mathematical capabilities of LaTeX. Standard graphics formats such as Encapsulated PostScript (EPS) or Scalable Vector Graphics (SVG) typically render text using fonts and formatting determined by the drawing application, which may not align with the document's overall aesthetic or may lack support for complex mathematical notation. To overcome this limitation, many tools, including AutoLaTeX, provide a mechanism for embedding TeX and LaTeX macros directly into the textual components of vector figures. When such a figure is processed, the textual elements are extracted, typeset by the LaTeX engine, and reintegrated into the final graphic, ensuring perfect harmony with the surrounding document.

With AutoLaTeX, the underlying principle rests on a class of translators that produce dual output: a PDF file containing the graphical elements (lines, shapes, images) and a companion TeX file that defines the textual overlays. The TeX file contains the macros necessary to position and typeset the text at precisely the coordinates intended by the figure's author. During document

compilation, the LaTeX engine reads this TeX file and renders the text using the same font settings, mathematics mode, and cross-referencing capabilities as the main document. This approach preserves the vector quality of the graphics while achieving seamless integration of LaTeX-typeset text.

Several translators within AutoLaTeX support this dual-output mode. For diagrams created with Xfig, the translator **fig2pdf+tex** processes files with extensions such as `+tex.fig`. The Xfig author must designate text objects that should be treated as LaTeX code by marking them in the editor (see documentation of Xfig). The translator extracts these objects, generates a separate TeX file, and produces a PDF of the graphic stripped of the original text. In Inkscape, the author creates text elements and applies the "LaTeX" rendering mode (via the "Text" menu or by setting the "LaTeX" output option in the export dialog). The translator then separates the graphical paths from the textual content, producing a PDF of the graphic and a `.pdftex_t` file containing the LaTeX code for the text.

Once the source figure has been prepared and saved with the appropriate extension, AutoLaTeX's translator system takes over. Upon detecting a change in the source file, the translator executes the necessary external commands (Inkscape, fig2dev, gnuplot, etc.) to generate the PDF and the TeX companion file. During the LaTeX compilation of the main document, these files are included using specialized macros. The `autolatex.sty` package provides `\includegraphicswtex{filename}`, which behaves like the standard `\includegraphics` but automatically incorporates the associated TeX macros from the companion file. The graphics path resolution for these macros is governed by `\DeclareGraphicsExtensionsWtex`, which by default includes `.pdftex_t`, `.pstex_t`, `.pdf_tex`, and `.ps_tex`.

The advantages of this approach extend beyond mere aesthetic consistency. Mathematical formulas, chemical notations, and custom LaTeX commands appear exactly as they do in the body text, with proper font selection, sizing, and spacing. References to equations, sections, or bibliographic items within figure labels become possible because the text is typeset at compilation time and can incorporate cross-references. Moreover, the resulting PDF graphics remain fully scalable and resolution-independent, preserving the crispness of both lines and text under zoom.

LATEX PACKAGES

AutoLaTeX provides two LaTeX packages called `autolatex.sty` and `autolatex-beamer.sty`. You could create a copy of these packages in the folder of your document by using the command-line's commands **createsty** and **createbeamer**.

`autolatex.sty`

The LaTeX package `autolatex.sty` provides the following TeX macros:

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

Includes a figure with combined TeX macros. The file on the file system must have a name with one of the extensions defined with `\DeclareGraphicsExtensionsWtex`. The `options` must be either `width=XX` or `height=XX`, where `XX` is a length.

`\includefigurewtex[options]{filename}`

Same as `\includegraphicswtex`.

`\includeanimatedfigure[options]{filename}`

Includes the layers of a figure in a Beamer presentation. The layers are assumed to be in separate PDF files. The figure is a TeX file that includes the PDF files in a Beamer-compatible environment. The file on the file system must have a name with one of the extensions defined with `\DeclareGraphicsExtensionsWtex`. The `options` must be either `width=XX` or `height=XX`, where `XX` is a length.
By default, a layer replaces the previous layer when displayed. You can change the overlay specification by adding the string `<spec>` to the layer title in your SVG editor. The `spec` part defines the slide numbers on which the layer appears in Beamer. For example, `<2>` means "only on slide 2"; `<6->` means "from slide 6 to the end."

`\includeanimatedfigurewtex[options]{filename}`

Includes the layers of a figure combined with TeX macros in a Beamer presentation. The layers are assumed to be in separate PDF files. The figure is a TeX file that includes the PDF files in a Beamer-compatible environment. The file on the file system must have a name with one of the extensions defined with `\DeclareGraphicsExtensionsWtex`. The `options` must be either `width=XX` or `height=XX`, where `XX` is a length.
By default, a layer replaces the previous layer when displayed. You can change the overlay specification by adding the string `<spec>` to the layer title in your SVG editor. The `spec` part defines the slide numbers on which the layer appears in Beamer. For example, `<2>` means "only on slide 2"; `<6->` means "from slide 6 to the end."

`\DeclareGraphicsExtensionsWtex{extensions}`

Defines the filename extensions used by `\includegraphicswtex` to locate figure files. The extensions may be separated by comma ',' characters. By default, the declared filename extensions are: `.pdftex_t`, `.pstex_t`, `.pdf_tex`, and `.ps_tex`.

`\graphicspath{{path1},{path2}...}`

This macro is from the TeX package 'graphicx.sty'. It defines the search paths for figures. You could specify multiple paths by separating them with coma ',' characters. It is recommended to enclose each path with block characters '{' and '}'. The figures' files are searched in the different search paths, in the order specified in the argument of `\graphicspath`.

`autolatex-beamer.sty`

The LaTeX package `autolatex-beamer.sty` provides the following TeX macros:

`\animatedfigureslide<frames>[options]{title}{filename}`

Create a frame/slide with the given title and that contains only a single figure that is an

animated figure according to the macros provided by the `autolatex.sty` package.
The `frames` parameter is optional and specify the frame numbers that must be displayed from the animated figure. Each frame corresponds to a layer from the animated figure. By default, a layer replaces the previous layer when displayed. You can specify the frames to be shown by adding the string `<frames>`. This `frames` part defines the slide numbers on which the layer appears in Beamer. For example, `<2>` means "only on slide 2"; `<6->` means "from slide 6 to the end."
The `options` must be either `width=XX`, `height=XX`, `label=ID` or `subtitle=TXT`, where `XX` is a length, `ID` is the identifier of the slide to be served as its label, and `TXT` is a regular text.
The `title` is used as the main title of the created slide.
The `filename` is the fielname or the path to animated figure. The file on the file system must have a name with one of the extensions defined with `\DeclareGraphicsExtensionsWtex`.

`\autolatexsettoslidecontentwidth{variable}`

This macro sets the value of the specified length variable (usually defined with a `\newlength`) to the *width of the text part* of a slide. This macro may be redefined by TeX Beamer templates for changing the computing of this width according to th template's graphical structure.

`\autolatexsettoslidecontentheight{variable}`

This macro sets the value of the specified length variable (usually defined with a `\newlength`) to the *height of the text part* of a slide. This macro may be redefined by TeX Beamer templates for changing the computing of this height according to th template's graphical structure.

CONFIGURATION FILE

Location of the Configuration Files

AutoLaTeX configuration files can be located in several places:

- **System Configuration** (for all users): In the directory where AutoLaTeX is installed (typically `/usr/lib/python3/dist-packages/autolatex2` on Unix systems).
- **User Configuration**: Two cases apply: either the configuration directory (`$HOME/.autolatex` on Unix or `C:\Documents and Settings\<user>\Local Settings\Application Data\autolatex` on Windows) exists, or it does not.
  In the first case, the configuration file is stored in the directory and named `autolatex.conf`.
  In the second case, the configuration file is in the user directory and named `$HOME/.autolatex` on Unix, or `C:\Documents and Settings\<user>\Local Settings\Application Data\autolatex.conf` on Windows.
- **Project Configuration**: The configuration file is in the same directory as the main TeX file of the document. It is named `.autolatex_project.cfg` on Unix and

`autolatex_project.cfg` on Windows.

Syntax of the Configuration Files

Configuration files follow a syntax similar to Windows .ini files.

Comments start with '#' or ';' and continue to the end of the line.

Each configuration directive must be within a configuration section, declared by its name in brackets (e.g., `[mysection]`).

Each directive is declared as: `directive name = value`

Several section names are reserved by AutoLaTeX; others are assumed to be translator configurations.

`[Generation]` section

This section configures the generation process used by AutoLaTeX. Recognized directives:

- `biber_cmd`: Specifies the Biber tool command line. Accepted value: any command line.
- `biber_flags`: Specifies options to pass to the Biber tool. Accepted value: any command line.
- `bibtex_cmd`: Specifies the BibTeX tool command line. Accepted value: any command line.
- `bibtex_flags`: Specifies options to pass to the BibTeX tool. Accepted value: any command line.
- `dvi2ps_cmd`: Specifies the dvips tool command line. Accepted value: any command line.
- `dvi2ps_flags`: Specifies options to pass to the dvips tool. Accepted value: any command line.
- `generate images`: Indicates whether AutoLaTeX automatically generates figures. Accepted values: `yes` or `no`.
- `generation type`: Indicates the type of generation. Accepted values:
  - `pdf` – generate a PDF document
  - `dvi` – generate a DVI or XDV document
  - `ps` – generate a PS document
- `image directory`: Specifies directories where AutoLaTeX finds pictures to be processed by translators. Paths are separated by the path-separator character (':' on Unix, ';' on Windows).
- `include extra macros`: Specifies if the definition of extra TeX macros (not from TeX or LaTeX standards) must be included and supported by AutoLaTeX.
- `latex_cmd`: Specifies the LaTeX tool command line. Accepted value: any command line.
- `latex_flags`: Specifies options to pass to the LaTeX tool. Accepted value: any command line.

- `main file`: Specifies the basename of the main TeX file to compile. This option is only available in the project's configuration file. See the section on extra macros for the details of the supported macros.
- `makeglossaries_cmd`: Specifies the MakeGlossaries tool command line. Accepted value: any command line.
- `makeglossaries_flags`: Specifies options to pass to the MakeGlossaries tool. Accepted value: any command line.
- `makeindex style`: Specifies the style to be used by MakeIndex. This is a comma-separated list of values in order of preference. Values include:
  - `<filename>` – if a filename is specified, AutoLaTeX assumes it is the .ist file;
  - `@system` – AutoLaTeX uses the system default .ist file (from the AutoLaTeX distribution);
  - `@detect` – AutoLaTeX attempts to find a .ist file in the project's directory. If none is found, no style is passed to MakeIndex;
  - `@none` – AutoLaTeX assumes no .ist file should be passed to MakeIndex;
  - `<empty>` – AutoLaTeX assumes no .ist file should be passed to MakeIndex.

    If the list contains multiple values, AutoLaTeX applies the corresponding behaviors in sequence.
- `makeindex_cmd`: Specifies the MakeIndex tool command line. Accepted value: any command line.
- `makeindex_flags`: Specifies options to pass to the MakeIndex tool. Accepted value: any command line.
- `synctex`: Indicates whether the output document should be produced with SyncTeX support.
- `tex compiler`: Indicates the TeX compiler to use. Accepted values:
  - `latex` – use `latex`
  - `pdflatex` – use `pdflatex`
  - `xelatex` – use `xelatex`
  - `lualatex` – use `lualatex`
- `translator include path`: Specifies additional directories from which translator scripts can be loaded. This is a list of paths separated by commas or the operating system's path separator (':' on Unix, ';' on Windows). If a path contains a comma, enclose it in quotes.

`[Clean]` section

This section configures the cleaning features of AutoLaTeX (targets `clean` and `cleanall`). Recognized directives:

- `files to clean`: A list of files to remove when the 'clean' target is invoked. Shell wildcards are allowed.
- `files to desintegrate`: A list of files to remove when the 'cleanall' target is invoked. Shell wildcards are allowed.

`[Scm]` section

This section configures the SCM support of AutoLaTeX (CVS, SVN, or others). Recognized directives:

- `scm commit`: Specifies the command line to use when committing changes.
- `scm update`: Specifies the command line to use when updating the local copy.

`[Viewer]` section

This section configures the viewer used by AutoLaTeX. Recognized directives:

- `view`: Indicates whether AutoLaTeX should launch a viewer after LaTeX compilation. Accepted values: `yes` or `no`.
- `viewer`: The path or command line of the viewer to launch. Accepted value: any command line.

Translator section

A translator section shares the name of the translator it configures. Recognized directives:

- `files to convert`: A list of files to be converted by this translator. Files are separated by the operating system's path separator (':' on Unix, ';' on Windows).
- `include module`: Indicates whether the translator should be loaded by default. Accepted values: `yes` or `no`.

EXTRA TEX AND LATEX MACROS

The recognition of the following TeX and LaTeX macros have been included in AutoLaTeX. These macros are not part of the standard TeX distributions. They are usually part of TeX templates provided by the authors of AutoLaTeX.

- `\animatedfigureslide[options]{title}{path}`: See above. Provided by AutoLaTeX.
- `\begin{bibliographysection}\end{bibliographysection}`: equivalent to `\begin{bibunit}\end{bibunit}` with a bibliography slide added at the end of the section with `\bibliographyslide`. Provided by the TeX-templates from https://github.com/gallandarakhneorg/tex-templates.
- `\bibliographyslide`: a Beamer slide with a Bibunits bibliography. Provided by the

TeX-templates from https://github.com/gallandarakhneorg/tex-templates.

- `\figureslide[options]{title}{path}`: a Beamer slide that contains a single figure. Provided by the TeX-templates from https://github.com/gallandarakhneorg/tex-templates.
- `\includeanimatedfigure[options]{path}`: See above. Provided by AutoLaTeX.
- `\includeanimatedfigurewtex[options]{path}`: See above. Provided by AutoLaTeX.
- `\includefigurewtex[options]{path}`: See above. Provided by AutoLaTeX.
- `\includegraphicswtex[options]{path}`: See above. Provided by AutoLaTeX.
- `\libraryslide[options]{picture}{title}{authors}{how published}{isbn}`: a Beamer slide that shows a description of a book. Provided by the TeX-templates from https://github.com/gallandarakhneorg/tex-templates.
- `\mfigure[position]{include graphics options}{filename}{caption}{label}[source text]`: Include a floating figure with the given arguments. Provided by the tex-upmethodology from https://github.com/gallandarakhneorg/tex-upmethodology.
- `\mfigure*[position]{include graphics options}{filename}{caption}{label}[source text]`: Include a floating figure with the given arguments. Provided by the tex-upmethodology from https://github.com/gallandarakhneorg/tex-upmethodology.
- `\msubfigure{filename}{caption}`: Include a subfigure with the given arguments in a floating figure. Provided by the tex-upmethodology from https://github.com/gallandarakhneorg/tex-upmethodology.
- `\msubfigure*{filename}{caption}`: Include a subfigure with the given arguments in a floating figure. Provided by the tex-upmethodology from https://github.com/gallandarakhneorg/tex-upmethodology.
- `\mfigurewtex[position]{include graphics options}{filename}{caption}{label}`: Include a floating figure embedding TeX macros and with the given arguments. The concept of figure with embedded TeX is based on the AutoLaTeX translators "+tex" as defined above. Provided by the tex-upmethodology from https://github.com/gallandarakhneorg/tex-upmethodology.
- `\mfigurewtex*[position]{include graphics options}{filename}{caption}{label}`: Include a floating figure embedding TeX macros and with the given arguments. The concept of figure with embedded TeX is based on the AutoLaTeX translators "+tex" as defined above. Provided by the tex-upmethodology from https://github.com/gallandarakhneorg/tex-upmethodology.
- `\partnerlogo{path}`: specification of a logo for an associated partner in Beamer template. Provided by the TeX-templates from https://github.com/gallandarakhneorg/tex-templates.
- `\resolvedfilename`: result of the resolution of a picture name. See `\resolvepicturename` below. Provided by the TeX-templates from https://github.com/gallandarakhneorg/tex-templates.
- `\resolvepicturename{name}`: search for a picture with the given basename inside one of the figure's paths and with one of the supported image filename extensions. The found filename for the picture is stored in the macro `\resolvedfilename`. Provided by the TeX-templates from https://github.com/gallandarakhneorg/tex-templates.
- `\sidecite{bibtex identifiers}`: in a Beamer template, add a note on the side of the slide that is equivalent to `\cite{bibtex identifiers}`. Provided by the TeX-templates from https://github.com/gallandarakhneorg/tex-templates.

- `\sidenote{text}`: in a Beamer template, add a text on the side of the slide. Provided by the TeX-templates from https://github.com/gallandarakhneorg/tex-templates.

BUG REPORT AND FEEDBACK

To report bugs, provide feedback, or suggest new features (in preferred order):

a) Visit the developer site on GitHub: https://github.com/gallandarakhneorg/autolatex2/

b) Visit the AutoLaTeX main page: http://www.arakhne.org/autolatex/

c) Send an email to the main author: galland@arakhne.org.

SYSTEM REQUIREMENTS

AutoLaTeX can be used directly from any directory where it is uncompressed, but you may want to compile and install additional files (manuals, etc.).

To configure and install AutoLaTeX, you may need Python 3.12 or higher and several Python packages, which are installed by the `setup.py` script.

To use AutoLaTeX, you will require:

- LaTeX. AutoLaTeX was developed using the TeX Live distribution.
- Python 3.12 or higher.
- Several Python packages, typically included in your Python distribution.

INSTALLATION

To install AutoLaTeX, run the `setup.py` script to compile and install AutoLaTeX. The basic commands are:

```
cd path_to_autoloader_sources/
./setup.py build install --optimize --install-layout deb
```

AUTOLATEX LICENSE

GNU Lesser General Public License (LGPL)

program; see the file `COPYING`. If not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

MANUAL COPYRIGHT

GNU Free Documentation License (FDL)

DOCUMENTED VERSION OF AUTOLATEX

This documentation is related to the version 51.1 of AutoLaTeX that was generated 2026-04-21.

SEE ALSO

`pdflatex`, `latex`, `bibtex`, `biber`, `epstopdf`, `fig2dev`, `gnuplot`, `inkscape`, `umbrello`, `zcat`, `texify`