

The packages **svg** and **svg-extract**

Philip Ilten (2012–2016)

Falk Hanisch (2017–)

<https://github.com/mrpiggi/svg>

hanisch.latex@outlook.com

v2.02d (2019/10/22)

The **svg** package is intended for the automated integration of SVG graphics into \LaTeX documents. Therefor the capabilities provided by **Inkscape**—or more precisely its command line tool—are used to export the text within a SVG graphic to a separate file, which is then rendered by \LaTeX . The two commands `\includesvg` and `\includeinkscape` are provided as central user-interface, which are very similar to the `\includegraphics` command of the **graphicx** package.

In addition, the package **svg-extract** allows the extraction of these graphics into independent files in different graphic formats, exactly as it is rendered within the document. For the creation of these graphics in the well-known formats PDF, EPS and PS, \LaTeX and possibly conversion tools shipped with the distribution are used. If the graphics are required in other file formats, either **ImageMagick** or **Ghostscript** can be invoked.

Contents

I.	User documentation	2
1.	Introduction	2
2.	Usage of package <code>svg</code>	3
2.1.	General settings	3
2.2.	Options for the invocation of Inkscape	4
2.3.	Options for the graphic inclusion	5
2.4.	Including SVG files	6
2.5.	Including already exported SVG files	6
3.	Usage of package <code>svg-extract</code>	7
3.1.	General settings	7
3.2.	Extract independent graphic files	8
3.3.	Convert extracted graphic files	9
3.3.1.	Settings for the invocation of ImageMagick	11
3.3.2.	Settings for the invocation of Ghostscript	11
4.	Example	11
5.	Troubleshooting and reporting issues	13
6.	Include SVG files created with <code>ROOT</code>	13
II.	Implementation	16
A.	Initialization	16
A.1.	Packages	16

A.2.	Helper macros	16
B.	Including SVG files with package <code>svg</code>	16
B.1.	Options	16
B.1.1.	The invocation of <i>Inkscape</i>	17
B.1.2.	Setting input folder and file	20
B.1.3.	Setting output folder	21
B.1.4.	Options for the inclusion of graphics	22
B.2.	Handling path information	23
B.3.	Optional Parameters for user commands	31
B.4.	User commands	31
B.5.	Auxiliary macros	35
B.6.	Patches	39
C.	Extracting independent graphic files with <code>svg-extract</code>	41
C.1.	Options	41
C.1.1.	Controlling the extract process	42
C.1.2.	Invoking external application for graphic conversion	45
C.1.3.	Setting output folder	48
C.1.4.	Options for the extraction of graphics	50
C.1.5.	Miscellaneous options	52
C.2.	User commands	53
C.3.	Auxiliary macros	54
C.4.	Commands for the separate auxiliary \LaTeX -file	66
D.	Processing Options	67
E.	Macros for file access	68
	Index	69
	Change History	72

Part I.

User documentation

1. Introduction

The open source program *Inkscape* has provided an excellent resource for the simple and easy creation of images and diagrams using a graphical user interface. The work by Johan B. C. Engelen has further enhanced the ability of *Inkscape* to split a SVG file into a text component that can be compiled with \LaTeX , and an image component that can be imported as a PDF file. For further information see the documentation of [svg-inkscape on CTAN](http://www.ctan.org/pkg/svg-inkscape)¹. The procedure described therein is taken up and consistently expanded. Thus, it is now possible to include a SVG file into a \LaTeX document where the text within the SVG graphic will be rendered natively by \LaTeX .

Both packages `svg` and `svg-extract` rely heavily upon executing commands from the shell using the `\ShellEscape` command—or respectively the old known `\write18`—for executing a variety of commands directly to the system. So it is necessary to include the flag `--shell-escape` when compiling documents using `svg` and/or `svg-extract`. The executed commands and the possibilities to adapt their invocation with the appropriate options are described later on in this documentation. All this is done automatically with the `\includesvg` command. If you don't want to use the `--shell-escape` flag, either for security reasons or because the export of the SVG files is done in another way, there's also the command `\includeinkscape` which includes files already exported by *Inkscape*.

¹<http://www.ctan.org/pkg/svg-inkscape>

An working installation of **Inkscape** is required for the automated integration of SVG graphics, whereby the installation path must be known to the operating system. This can be checked on shell by typing `inkscape -V`. Moreover, there are some required packages which are loaded by packages **svg** and **svg-extract** to provide the functionality. These are:

scrbase for the definition and handling of options in key-value-syntax
ifpdf, ifluatex, ifxetex for flow control depending on the used L^AT_EX engine
pdfdoccmds, shellesc to allocate the same primitives independent of the used L^AT_EX engine
ifplatform to control the file access depending on the operating system
trimspaces to remove unwanted spaces in file paths
graphicx for including the graphic files after the **Inkscape** export
xcolor,transparent are possibly needed by the separate L^AT_EX files created by **Inkscape**
xr is used by **svg-extract** in order to include labels within the independent graphic files

If you want to pass options to package **graphicx**, you must either load it before package **svg**

```
\usepackage[options]{graphicx}
...
\usepackage[options]{svg}
```

or use `\PassOptionsToPackage`.

```
\PassOptionsToPackage{options}{graphicx}
...
\documentclass[options]{class}
...
\usepackage[options]{svg}
```

The usage of packages **xcolor** and **transparent** can be switched off while loading package **svg**. See the two options `usexcolor` and `usetransparent` below.

2. Usage of package `svg`

The purpose of this package is to include SVG graphics into a L^AT_EX document. The command `\includesvg` is defined which does all necessary steps for this task. It first launches the export of a SVG file to a supported file format with Inkscape, if necessary, and includes the exported graphic file afterwards. The usage and the syntax is quite similar to the command `\includegraphics` from the **graphicx** package. In fact, the inclusion of the exported graphic file is done with `\includegraphics`.

`usexcolor` (opt.) The packages **xcolor** and **transparent** are loaded by default at the end of package **svg**. The listed options are intended to prevent these packages from loading. They are the only options which have to be given while loading the **svg** package. All supported boolean values (`true/on/yes/false/off/no`) can be assigned to `usexcolor` and `usetransparent`, while `noxcolor` and `notransparent` don't accept any value.

`usetransparent` (opt.)

`noxcolor` (opt.)

`notransparent` (opt.)

```
\usepackage[options]{svg}
```

2.1. General settings

`\svgsetup` All other options described in detail below can also be changed after loading the package either in the preamble or within the document. They don't have to be given as optional argument to `\usepackage[options]{svg}` but can be set by using macro `\svgsetup{options}` where `{options}` is a comma separated list of options. Settings with `\svgsetup` are done in the current scope which means globally or within the current group.

```
\svgsetup{options}
```

Further, it's possible to reset any setting locally with the optional argument of the commands `\includesvg[options]{svg filename}` or `\includesvg[options]{graphic filename}`.

`\svgpath` Most likely you want to organize your SVG files in a separate folder either as a subfolder in the working directory or elsewhere in your local folder structure. For this purpose, a list of root paths to SVG files can be specified using the `\svgpath` command in the same way as `\graphicspath` is used. Every path has to be given in a group of braces `{}`—even if there is only one—and terminate with `/` last. For example:

```
\svgpath{{svg/}{usr/local/svg/}}
```

would cause the system to look first in the subdirectory `svg/` and afterwards in the absolute path `/usr/local/svg/`. Further, if no path was specified with `\svgpath` or the desired file wasn't found, all directories given with `\graphicspath` are searched too. Please keep in mind that the current working directory is browsed first in any case. It's recommended to avoid any spaces and/or quotes respectively `\dq` both in paths and file names, especially when DVI output is active.

2.2. Options for the invocation of *Inkscape*

`inkscape` (opt.) This option controls, when the export with *Inkscape* is invoked and is `true` by default.

`false/off/no`

Inkscape won't be invoked in any case, no export is done.

`true/on/yes/newer/onlynewer`

The export with *Inkscape* will only be done, if the exported graphic file either does not exist or the file modification date of the SVG file is newer than that of the exported graphic file. Thus the compilation time of the \LaTeX document can be reduced to the necessary minimum.

`forced/force/overwrite`

The *Inkscape* export will definitely be done, any already existing exported file will be overwritten regardlessly.

In addition to controlling the export behavior, the option `inkscape` can also be used to make additional settings, which then acts as a wrapper for the options described below.

`pdf/eps/ps/png`

see `inkscapeformat=pdf/eps/ps/png`

`latex/nolatem`

see `inkscapelatem=true/false`

`drawing/page`

see `inkscapearea=drawing/page`

`<integer>dpi`

see `inkscapedpi=<integer>`

`inkscapepath` (opt.) The option `inkscapepath` specifies, where the resulting files of the *Inkscape* export should be located. The subfolder `./svg-inkscape/` within the current working directory is used by default (`inkscapepath=basesubdir`).

`svgdir/svgpath`

The PDF/EPS/PS/PNG graphic files as well as the \LaTeX files generated by *Inkscape* will be located in the same directory as the corresponding SVG file.

`svgsubdir/svgsubpath`

Within the folder of the encountered SVG file, all exported files will be located in a subfolder named `svg-inkscape/`.

`basedir/basepath/jobdir/jobpath`

All exported files will be located in the current working directory.

`basesubdir/basesubpath/jobsubdir/jobsubpath`

A subfolder named `svg-inkscape/` within the current working directory will be used for files generated by *Inkscape*.

`/path/to/somewhere/`

It is also possible to give a custom path, either relative to the current working directory (`./relative/path/`) or as an absolute path.

- `inkscapefilename` (opt.) The file names of the **Inkscape** export are derived from the name of the base SVG file and can be modified with `inkscapefilename=<filename>`. It's possible to use counters for specifying the name of the exported file. Repeatedly specifying the same file name will overwrite previously created files.
- `inkscapeexe` (opt.) For including a SVG file, **Inkscape** is used to separate the text and image from the SVG file itself. In order to execute the command line tool from shell, the path where the executable is located has to be known to the operating system. You can check this by typing `inkscape -V` into the shell. If this check fails and you don't want to change environment variable `path` on your OS, you can use option `inkscapeexe` to set the absolute path where the executable of **Inkscape** is located. The option is set to `inkscapeexe=inkscape` by default.
- `inkscapeformat` (opt.) With this option, the **Inkscape** export format can be controlled. Valid values are `pdf`, `eps`, `ps` and `png`, where a \LaTeX export is not possible for `png` and option `inkscapelatex` won't have any effect. By default, `inkscapeformat=pdf` is set unless DVI output was detected. In this case `inkscapeformat=eps` is the default setting.
- `inkscapelatex` (opt.) If option `inkscapelatex=true` is set, the output is split into a separate PDF/EPS/PS file (see option `inkscapeformat`) and a corresponding \LaTeX file. This is the default setting. Setting `inkscapelatex=false` will result in a single PDF/EPS/PS file, where any contained text won't be rendered by \LaTeX .
- `inkscapearea` (opt.) This option controls which area of the SVG file should be exported, `drawing` is set by default.
- drawing/crop**
The area exported corresponds to the bounding box of all objects in a drawing, including any that are not on the page.
- page/nocrop**
The area exported will correspond to the defined page area within the SVG file.
- `inkscaledpi` (opt.) The resolution used either for PNG export or for fallback rasterization of filtered objects when exporting to PDF/EPS/PS file. For PNG export it is set to 300 dpi by default, if no value was given. The given value should be a positive integer. The default behaviour can be reversed after a given value with `inkscaledpi=\relax`.
- `inkscapeopt` (opt.) You can use this option to pass additional switches to the **Inkscape** command line tool. For further information see the documentation of **Inkscape**².
- `svgextension` (opt.) The package assumes SVG files with `.svg` extension as source for the **Inkscape** export. This option can be used to change this behaviour. For example, in order to process `.dia` files instead of `.svg` you could use

```
\includesvg[svgextension=dia,<additional options>]{<filename>}
```

2.3. Options for the graphic inclusion

- `width` (opt.) The width of the included graphic file can be specified via the `width` option and the height
- `height` (opt.) by the `height` option. If both the width and height are specified, the figure will be scaled
- `distort` (opt.) such that neither of the specified dimensions is exceeded, unless option `distort=true` is
- `scale` (opt.) given.³ If `width` and/or `height` once have been set, this can be undone by setting them to `Opt` or `\relax`. If neither `width` nor `height` are set, the included graphic file can also be scaled by setting `scale` to a positive real number.
- `pretex` (opt.) Commands prior and post to the inclusion of the graphic file may be desired, such as font or
- `aptex` (opt.) color commands. The options `pretex` and `aptex` are provided where the \LaTeX code given to `pretex` is included before the graphic file and `aptex` right afterwards. For example, to change the size of the included text one could use:

```
\includesvg[pretex=\tiny,<additional options>]{<svg filename>}
```

²<https://inkscape.org/de/doc/inkscape-man.html>

³to provide compatibility for package `graphicx`, it's possible to use `keepaspectratio=true` as alias for `distort=false` and the other way round

- `draft` (opt.) This option can be used with boolean values and is equal to the identically named option of the **graphicx** package. If the `draft` option is given to **graphicx**, it's activated for **svg** as well.
- `lastpage` (opt.) A [bug](#)⁴ concerning the L^AT_EX export has been reported for **Inkscape** 0.91. It may happen that within the exported L^AT_EX file, it's attempted to include more pages of the PDF graphics than actually exist. The **svg** package attempts to bypass the resulting error.

Consequently, the total number of pages is read and only existing PDF pages are included, if both options `inkscapeformat=pdf` and `lastpage=true` are set. This is the default setting and can be switched off with `lastpage=false`. It's also possible to set the number of the last page included of a PDF graphic manually as optional parameter for `\includesvg` or `\includeinkscape`. For details, see the description of the respective commands.

2.4. Including SVG files

- `\includesvg` The command `\includesvg` to include a SVG file is quite similar to the `\includegraphics` command provided by the **graphicx** package.

```
\includesvg[parameters]{svg filename}
```

- `inkscape` (param.) It is used right in the same way but where `{svg filename}` is the file name of the SVG file, where any given file extension will be replaced with `.svg` ruthlessly. In order to change the source file format for the **Inkscape** export, you have to use parameter `svgextension`.
- `inkscapeformat` (param.)
- `inksapelatex` (param.)
- `inkscapearea` (param.)
- `inkscapedpi` (param.)
- `inkscapeopt` (param.)
- `svgextension` (param.) If the given file is not located in the current working directory but elsewhere on your file system, the command `\svgpath` could be used to specify this path. It is recommended to avoid any spaces and/or quotes respectively `\dq` both in paths and file names. Especially when DVI output is active using quotes will certainly cause an error.
- `width` (param.)
- `height` (param.)
- `distort` (param.) The command `\includesvg` is intended to do an automated export with **Inkscape** at first, where the given SVG file is exported to a PDF/EPS/PS/PNG file (see `inkscapeformat`) and perhaps a correlating L^AT_EX file (see `inksapelatex`). The export with **Inkscape** is only invoked, if the SVG file is newer than the exported graphic file or latter doesn't exist at all. Once the export has been done, the graphic file and maybe the L^AT_EX file are included.
- `scale` (param.)
- `pretex` (param.)
- `apptex` (param.)
- `draft` (param.) All previously described options can also be used as optional parameters to `\includesvg` and do have the same effect as described before. However, the optional parameters specified have an effect only once when `\includesvg` is executed and remain unchanged afterwards.
- `lastpage` (param.) In addition to the use of boolean values, the parameter `lastpage` can also be assigned a specific (integer) page number, which defines the last used page of a PDF graphic. This, just like the identically named option, has an effect only when `inkscapeformat=pdf` is set.
- `angle` (param.)
- `origin` (param.) Both parameters correlate to the identically named parameters of the `\includegraphics` command provided by the **graphicx** package. However, unlike to `\includegraphics`, they `angle` and `origin` are *always evaluated after width, height, distort and scale* by `\includesvg`, regardless of the used order of the given parameters. This is mainly due to the inclusion of the L^AT_EX files corresponding to the graphic files generated by **Inkscape**.

2.5. Including already exported SVG files

- `\includeinkscape` If you don't want to make use of the automated export with **Inkscape** but the user interface provided by the **svg** package, you can use `\includeinkscape` instead of `\includesvg`.

```
\includeinkscape[parameters]{graphic filename}
```

- `inkscapeformat` (param.) You can use it similar to `\includesvg` but `{graphic filename}` has to be the filename of the already exported graphic file. If a valid file extension (`.pdf/.eps/.ps/.png`) is given, the current setting for `inkscapeformat` is overwritten. It's even possible to specify a file extension like `.pdf_tex` to activate `inksapelatex`. Furthermore, all optional parameters for `\includeinkscape` do have the same effect as described before for command `\includesvg` once when `\includeinkscape` is executed and remain unchanged afterwards.
- `inksapelatex` (param.)
- `width` (param.)
- `height` (param.)
- `distort` (param.)
- `scale` (param.)
- `pretex` (param.)
- `apptex` (param.)
- `draft` (param.)
- `lastpage` (param.)
- `angle` (param.)
- `origin` (param.)

⁴<https://bugs.launchpad.net/ubuntu/+source/inkscape/+bug/1417470>

3. Usage of package `svg-extract`

This package allows the extraction of independent graphic files out of SVG files which have been included and rendered with L^AT_EX by the `svg` package. This is particularly useful when attempting to provide images to journals or collaborators, and one wishes the image to appear exactly as it does within the original L^AT_EX document.

In order to extract to PDF, EPS, or PS files the programs `pstoeps`, `pstopdf` and `pdftops` are used which are usually provided by most of the L^AT_EX 2_ε distributions. In addition, the command line tools of *ImageMagick* and *Ghostscript* can be invoked for converting images in formats like PNG, JPG, TIF or something else. It's also possible to create PDF, EPS or PS files with one of the two programs. Therefor the desired program—`magick` and/or `gswin32c/gswin64c` on Windows respectively `convert` and/or `gs` on unix-like operating systems—must be installed. By typing `<program> --version` on shell, this can be checked.

If you want to extract independent graphic files from included SVG files, you only have to load `svg-extract`. All actions for the extraction process will be done by using `\includesvg` or `\includeinkscape`. Without any additional settings, the extraction will render the SVG file to the specified output formats(s) of choice using the same settings as specified within the two commands. Consequently, the scale between the image and text in the extracted files will remain identical to the scale within the document from which the SVG file was extracted.

In contrast to package `svg`, the console commands for graphic extraction are executed with each LaTeX run by package `svg-extract` when `--shell-escape` mode is activated. This behaviour can be switched of with option `extract=false`.

Important changes

In version v1.0 of package `svg` the extracted files were named like the numbering of the current `subfig` environment by default. As package `subfig` sometime causes problems and because of the large amount of different L^AT_EX packages which all provide the possibility to include subfigures with very different implemetations, this feature can't be provided reliably by `svg-extract`. See option `extractname` for further information.

3.1. General settings

`on` (opt.) This options have to be given while loading the `svg-extract` package and are intended to toggle the functionality of this package. As both extracting and converting independent graphic files is invoked with every L^AT_EX run when `--shell-escape` is activated, the option `off` can be given to save compilation time, once the creation of all desired images has been done and they no longer need to be re-generated. The option `on` can be used to reactivate functionality of this package. This can also be done by using `extract=true/false`.

`\svgsetup`
`\includesvg`
`\includeinkscape` With package `svg-extract` the applicable options for `\svgsetup{<options>}` as well as parameters for the already described macros `\includesvg[<parameters>]{<filename>}` and `\includeinkscape[<parameters>]{<filename>}` are extended. They can be used to control the process of graphic extraction and converting.

`extractangle` (param.) With this parameter the graphic is rotated during the extraction process. The value is not inherited from `angle` if it was given by default. this can be achieved by setting:

```
\includesvg[angle=<angle>,extractangle=inherit]{<filename>}
```

All option described below can be used together with `\svgsetup` and are then valid in the current scope. There also exist identically named parameters for the optional arguments of

```
\includesvg[<parameters>]{<svg filename>}  
\includeinkscape[<parameters>]{<graphic filename>}
```

These parameters have an effect only once when the specific command is executed and remain unchanged afterwards. These parameters are: `extract`, `extractpreamble`, `extractformat`, `extractruns`, `latexopt`, `extractwidth`, `extractheight`, `extractdistort`, `extractscale`, `extractangle`, `extractpretex`, `extractapptex`, `convert`, `convertformat`, `convertdpi`, `magicksetting`, `magickoperator`, `gsopt`, `gsdevice`, `clean`, `exclude`.

3.2. Extract independent graphic files

- `extract` (opt.) This option can be used with boolean values. Using `extract=true` activates the functionality for both extracting and converting which is the default setting, whereas `extract=false` turns it off completely.
- `extractpath` (opt.) The path where the extracted and converted files are located can be specified with option `extractpath`, whereas `extractpath=basesubdir` is set by default.
- `svgdir/svgpath`
The extracted and converted independent graphic files are located in the same directory as the corresponding SVG file.
- `svgsubdir/svgsubpath`
Within the folder of the encountered SVG file, all extracted and converted files will be located in a subfolder named `svg-extract/`.
- `basedir/basepath/jobdir/jobpath`
All extracted and converted files will be located in the current working directory.
- `basesubdir/basesubpath/jobsubdir/jobsubpath`
A subfolder named `svg-extract/` within the current working directory will be used for all extracted and converted files.
- `/path/to/somewhere/`
It is also possible to give a custom path, either relative to the current working directory (`./relative/path/`) or as an absolute path.
- `extractname` (opt.) It's also possible to change the name for extracted and converted files. The default setting is `extractname=filenamenumbered`.
- `filename/name`
The name of the exported *Inkscape* file is used and the suffix `-extract` is attached.
- `filenamenumbered/namenumbered/numberedfilename/numberedname`
Same as above, but a prefix with the count of extracted files is used instead of the suffix.
- `numbered/section/numberedsection/sectionnumbered`
The file name is composed by the number of extracted files and the current outline numbering.
- `{filename}`
You can use any file name, the file extension is derived from option `extractformat`. It's possible to use counters for specifying the name of the extracted file. Repeatedly specifying the same file name will overwrite previously created files.
- `extractformat` (opt.) The included SVG file can be extracted from the document into a independent graphic file of type PDF, EPS or PS. The option can be used with either a single value (`extractformat=pdf`) or a comma separated list. For example,
- ```
\includesvg[extractformat={pdf,eps,ps}]{<svg filename>}
```
- will extract the SVG file to both PDF and EPS formats and generates two independent graphic files. By default, `extractformat=pdf` is set unless DVI output was detected. In this case `extractformat=eps` is the default setting.
- `extractwidth` (opt.) These options can be used to overwrite the settings given for the appearance of a SVG file within the document. For example, a SVG file should cover the entire text width within the document but be extracted to a fixed width, this can be done with:
- ```
\includesvg[width=\textwidth,extractwidth=500pt]{<svg filename>}
```
- `extractheight` (opt.)
- `extractdistort` (opt.)
- `extractscale` (opt.)
- `extractpretex` (opt.)
- `extractapptex` (opt.) Assigning the value `inherit` to one of these options—which is set by default—leads to the usage of the corresponding option of package `svg` (`width/height/scale/pretex/apptex`), whereas `extract...=\relax` can be used to ignore a parent option utterly. Only option `extractdistort` is initialized to `false` and does not inherit from `distort` by default.
- `extractpreamble` (opt.) Within the included and extracted SVG files any \LaTeX macro can be used either defined by the user—this should be done in the preamble of the \LaTeX document in which the SVG file is to be included—or provided by a package which is loaded. As the extraction process of the SVG files needs an auxiliary \LaTeX file all used packages and commands have to be
- `extractpreambleend` (opt.)

known within this file. Consequently, the preamble of the current L^AT_EX document is used for the extraction of the SVG file by default.

However, it is possible to specify a different *preamble file* with the option `extractpreamble` where the file to use as the preamble is given as the argument—including maybe path, but file name and file extension in any case. The given preamble file is searched similar to SVG files meaning, every path given with `\svgpath` or `\graphicspath` is examined. The default definition of `extractpreamble` is `\jobname.tex`—more precisely the file extension given by option `latexext` is used—and should suffice for most cases. The preamble up to the line defined by the option `extractpreambleend` will be used, which is set to a default with `\begin{document}`.

`\svghidepreamblestart` In case, the preamble of the current L^AT_EX document is used, there are maybe packages included or some parts within the preamble, which should not be used within the separate auxiliary L^AT_EX file. These parts can be excluded if they are enclosed by `\svghidepreamblestart` and `\svghidepreambleend`.

For example, your current L^AT_EX document uses package `showframe` which causes some problems with the extraction of independent graphic files. So you want to get rid of it within the auxiliary L^AT_EX file. This can be done with:

```

\documentclass{<documentclassname>}
...
\usepackage{svg-extract}
...
\svghidepreamblestart
\usepackage{showframe}
\svghidepreambleend
...

```

`extractruns` (opt.) When extracting independent graphic files by compiling the generated auxiliary L^AT_EX file, it's maybe necessary to do multiple L^AT_EX runs on this file. The number of runs can be controlled with option `extractruns`. It's set to `extractruns=2` by default.

`latexexe` (opt.) For the extraction of an independent graphic file, the L^AT_EX program is used which is set by the `latexexe` option. Depending on the L^AT_EX processor used for the current L^AT_EX document, `latexopt` (opt.) it is set to either *pdflatex*, *lualatex*, *xelatex* or *latex* by default. It's also possible to specify additional flags or switches for the L^AT_EX runs, which are performed during the extraction process by the `latexopt` option. If you are used to utilize a other file extension for L^AT_EX files than `.tex`, option `latexext` can be used like `latexext=ltx`.

`dvipsopt` (opt.) Depending on the used L^AT_EX processor, the file type of the extracted graphic differs. In order to create all formats, requested with option `extractformat`, several converting tools `pstoeps` (opt.) provided by most of the L^AT_EX 2_ε distributions are maybe invoked. These are *dvips*, *ps2eps*, `pstopdf` (opt.) *ps2pdf* and/or *pdftops* and can't be changed. It's only possible to specify additional `pdftops` (opt.) switches for every single tool with `dvipsopt`, `pstoeps`, `pstopdf`, `pdftops` and `pdftops`.

`clean` (opt.) During the extraction process many files are generated for each SVG file extraction. So it's oftentimes desirable to automatically remove these temporary files. Using the option `clean=true` will remove any generated files created other than the extracted output format(s) requested. Setting `clean=false` is useful for debugging and set by default. Additionally, it's possible to use option `clean` with a list of file extensions in order to specify auxiliary files generated by package `svg-extract` to be deleted, for example `clean={log,aux}`.

`exclude` (opt.) Sometimes it may be necessary to extract and/or convert a SVG file without including it. If the flag `exclude` is specified, the SVG file will not be rendered in the current L^AT_EX document, but will be extracted and/or converted to the requested output format(s).

3.3. Convert extracted graphic files

Based on the extraction of independent graphic files, the `svg-extract` packages also provides the possibility to convert those extracted graphics in another format than PDF, EPS or PS with either *ImageMagick*—which is set by default—or *Ghostscript*.

`convert` (opt.) This option can be used to control the invocation of the conversion process. By default, `convert=false` is set. For Windows, there exist two different versions of **Ghostscript**, either 64 bit or 32 bit. If it is selected as converting tool the 64 bit executable is set by default.

`false/off/no`

No conversion is done.

`true/on/yes`

The conversion will be done with the current chosen converting tool.

`magick/imagemagick/convert`

The conversion is activated and **ImageMagick** is selected.

`gs/ghostscript`

The conversion is activated and **Ghostscript** is selected.

`gs64/ghostscript64`

This value activates **Ghostscript** as conversion tool and sets `gsex=gswin64c`. On unix-like operating systems, the value for `gsex` remains unchanged.

`gs32/ghostscript32`

The same as for the latter case applies, only option `gsex=gswin32c` is set on Windows.

`convertformat` (opt.) With this option, the desired output format(s) can be given. Multiple graphic formats can be specified in a list, for example something like `convertformat={png,jpg,tif}`. The value specified in `extractformat` is used as the source format for the conversion. If `extractformat` itself contains a file list, the first value within this list is considered. If `extractformat` is defined empty, the file generated anyway during the extraction is used.

Settings for specific converting formats

Maybe it's desired to apply varying settings for different output formats. Therefore some options described below can either be set for all converted files or for a specific output format. In particular, these are the options `convertdpi` as well as `magicksetting`, `magickoperator`, `gsdevice` and `gsopt`. All these mentioned options can be used like either `<option>=<value>` or `<option>={<outputformat>=<value>}` and even `<option>={<outputformat>+=<value>}` where the desired output format is trailed with + as inner key.

The first variant is applied to all output formats in general. If one of these mentioned options is evaluated and a output format specific value was given like in the second variant, the general setting is overwritten. If the general setting should be used and extended by an additional output format specific settings, then the third variant is to be used. In this case, no output format specific setting (second variant) must not have been used.

If you want to reverse any setting, you only have to use `\relax` as a value, either for a general option (`<option>=\relax`) or a specific one (`<option>={<outputformat>[+]=\relax}`).

`convertdpi` (opt.) This option controls the used density for all file formats or a specific one, whether **ImageMagick** or **Ghostscript** is used for the graphic conversion. The desired resolution of the converted file is given in dots per inch (DPI) either as a scalar value (e.g. `convertdpi=600`) or with different resolutions in x- and y-direction (e.g. `convertdpi=600x400`).

As described before, it's also possible to declare a specific resolution for each desired converting format. For example, you want to set different resolution for PNG and JPG formats and something for all other formats:

```
\svgsetup{%
  convertdpi={png=600},%
  convertdpi={jpg=150},%
  convertdpi=300%
}%
```

If a setting for a specific output format is given, any unspecified setting is overwritten, when the conversion to this format is done. With `convertdpi={<outputformat>=\relax}` a specific setting can be reversed.

Please note that not every graphic format support different resolutions in x- and y-direction. So using a value like `convertdpi=600x400` may not necessarily lead to the desired result. However, this is then due to the used conversion tool and not to the processing of the option.

3.3.1. Settings for the invocation of *ImageMagick*

`magickexe` (opt.) The conversion with *ImageMagick* via the `magick` or `convert` command-line tool can be controlled with these options. The option `magickexe` determines the used executable and is set to `magick` on Windows and otherwise to `convert` by default. Additionally, there are the two options `magicksetting` and `magickoperator` which can be used to define *settings* and *operators* for the conversion process. As described before, the two options `magicksetting` and `magickoperator` can be set for all output formats or a *specific* one either resetting or extending the general settings. For further information see the documentation of *ImageMagick* [command-line tool](#)⁵.

3.3.2. Settings for the invocation of *Ghostscript*

`gsexe` (opt.) The conversion with *Ghostscript* is done with command-line tool `gs` on unix-like operating systems and `gswin64c` or `gswin32c` on Windows. The executable can be changed with option `gsexe`. Because *Ghostscript* requires the specification of a device, there are some predefined for the most common output formats. These are:

```
\svgsetup{%
  gsdevice={png=png16m},gsdevice={jpeg=jpeg},gsdevice={jpg=jpeg},%
  gsdevice={tif=tiff48nc},gsdevice={tiff=tiff48nc},%
  gsdevice={eps=eps2write},gsdevice={ps=ps2write}%
}%
```

Furthermore, with `gsopt` additional switches for *Ghostscript* can be set. As described before, both `gsdevice` and `gsopt` can be defined in general or for specific output formats. For further information see the documentation of *Ghostscript*⁶.

4. Example

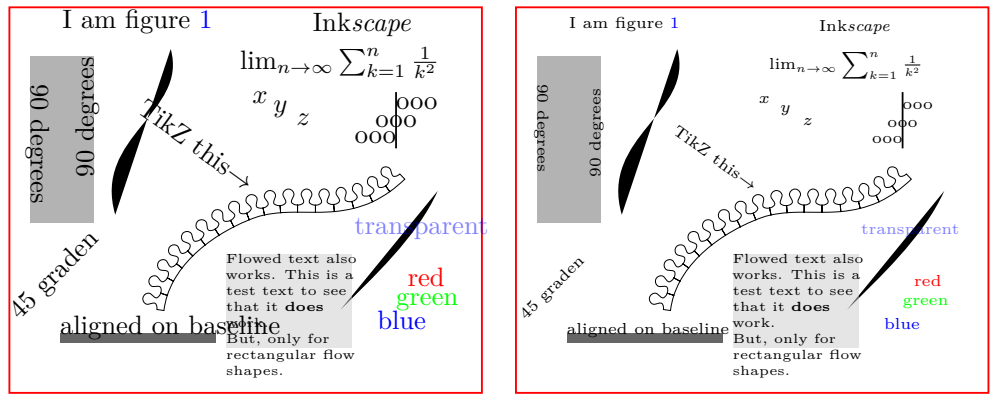
As an minimal example⁷ take the following lines of code:

```
\documentclass{article}
\usepackage[T1]{fontenc}
\usepackage{svg}
\usepackage[off]{svg-extract}
\svgsetup{clean=true}
%\pdfsuppresswarningpagegroup=1
\usepackage{relsize}
\usepackage{subcaption}
\begin{document}
\begin{figure}
\begin{minipage}{.5\linewidth}
\includesvg[width=\linewidth]{svg-example}%
\subcaption{This text is too large!}
\end{minipage}%
\begin{minipage}{.5\linewidth}
\includesvg[width=\linewidth,pretex=\relscale{0.6}]{svg-example}%
\subcaption{This text fits better.}
\end{minipage}
\caption{An example figure with \LaTeX-support}\label{fig:example}
\end{figure}
\begin{figure}\centering
\includesvg[%
width=.5\linewidth,inkscapelatex=false,extractformat={pdf,eps}%
]{svg-example}%
\caption{The same example figure without \LaTeX-support}
\end{figure}
```

⁵<http://www.imagemagick.org/script/command-line-processing.php>

⁶<https://ghostscript.com/doc/current/Use.htm>

⁷The image used here is a slightly modified version of the image used in the initial documentation on how to include a SVG file in L^AT_EX by Johan B. C. Engelen available as package [svg-inkscape](#) on CTAN.



(a) This text is too large!

(b) This text fits better.

Figure 1: An example figure with \LaTeX support

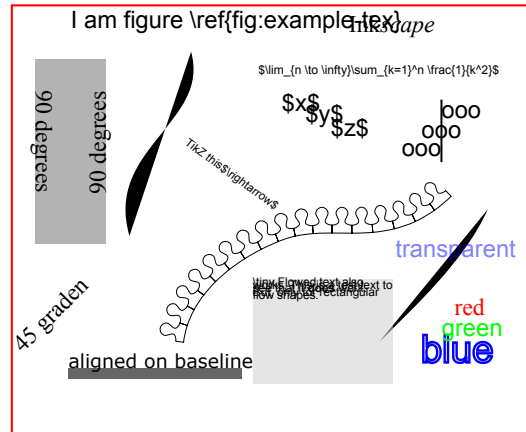


Figure 2: The same example figure without \LaTeX support

```
\end{document}
```

If you are willing to compile the example, there are two aspects to consider. First, the included SVG file `svg-example.svg` has to be located in the current folder and is located in `<texmf>/doc/latex/svg/examples/`. Second, you have to run the desired \LaTeX engine with `--shell-escape` option enabled.

The output is shown in [Figure 1](#) and [Figure 2](#). Within this example the file `svg-example.svg` was included three times using the `\includesvg` command.

As you can see, [Figure 1a](#) is created with default settings, except for the width specification. So the **Inkscape** export with \LaTeX support is done as well as the extraction of a independent graphic file in PDF format as the **svg-extract** package was loaded.

However, the text is slightly overrunning the margins of the image, and so [Figure 1b](#)—which again uses the same **Inkscape** export results—decreases the font size of the text within the image relative using the `pretex` option together with the `\relscale` command provided by the **relsize** package.

In [Figure 2](#) the same SVG file was used but without the export of a separate \LaTeX file containing all text elements.

Feel free to use this given example to try out all the options and possibilities described in [section 2](#) for package **svg**. Especially if you want to use package **svg-extract** for the automated extraction of independent graphics ([subsection 3.2](#)) and their conversion to different graphic formats with **ImageMagick** and/or **Ghostscript** ([subsection 3.3](#)), this example can be easily used for the first steps.

5. Troubleshooting and reporting issues

When using the packages **svg** and **svg-extract**, the most likely occurring problems will be caused by calling the external programs. For this reason, a short package information is written into the log file right before each call of an external program via shell. If a file should have been created, both packages check after the external call, whether this file exists or not and raise an error or at least a warning, if this file is missing. If you got such a message, please check the log file for lines like:

```
Package svg Info: or Package svg-extract Info:
```

Right afterwards, there should appear `runsystem(<command>)...executed.` which you should try to execute manually from shell in the right directory. In most cases, the problem will be an invalid command call. If something goes wrong during the extraction/converting process of package **svg-extract**, it would make sense to set option `clean=false` to not delete any auxiliary files that might be needed.

If you are sure that the problem is not caused by the configuration of your operating system, you can send an error report either via email or create a new issue on GitHub. Both addresses can be found on the title.

When using pdfL^AT_EX there are a lot of warnings

It may happen that several warnings like

```
pdfTeX warning: pdflatex.exe(file <filename>.pdf): PDF inclusion:
multiple pdfs with page group included in a single page
```

occur when including the PDF graphics exported with **Inkscape**. This is related to the handling of transparency effects within PDF files. Since pdfL^AT_EX version 1.40.15 or later, you can get rid of these messages by using `\pdfsuppresswarningpagegroup=1`. See also the discussion on [LaTeX Stack Exchange](#)⁸ for more information.

6. Include SVG files created with **ROOT**

This section was originally written by Philip Ilten. In the hope that since then nothing has changed fundamentally in the described procedure, this passage remains in the documentation, even if it will almost certainly be relevant to experimental particle physicists only, who frequently use the analysis package **ROOT**.

ROOT has the ability to export directly to a SVG file, which means that it is possible to completely by-pass all of **ROOT**'s internal text rendering machinery, and let L^AT_EX handle the text natively. This means that all of the ugly fonts that are rendered by **ROOT** can now be completely avoided, with the additional bonus of being able to add references within plots. So how does one go about using this package with **ROOT**?

1. Create the plot with **ROOT** as normal, but turn off all L^AT_EX interpretation of text strings. This is a bit tricky, but can be accomplished by setting the font in **ROOT** to a precision of zero as described in the documentation for **TAttFill**⁹. Remember that the font is set by using the function `(TAttFill*)->SetTextFont(i)` with

$$i = (\text{font type}) \times 10 + (\text{font precision})$$

In the following lines of code, a `TStyle` is defined which sets the font to type “Courier New” with a precision of zero.

⁸<http://tex.stackexchange.com/questions/76273/>

⁹<http://root.cern.ch/root/html/TAttText.html>

```
TStyle *style = new TStyle("style","style"); int FONT = 80;
style->SetTextFont(FONT);
style->SetLabelFont(FONT,"XYZ");
style->SetTitleFont(FONT,"XYZ");
style->SetTitleFont(FONT,"");
gROOT->SetStyle("style");
gROOT->ForceStyle();
```

Now, you can just use the well-known standard \LaTeX syntax for creating labels, etc. Note however, that backslashes have to be escaped due to interpretation of special characters by *C++*.

2. Print the plot as a SVG file.

```
gPad->Print("foo.svg");
```

3. Include the SVG file within the document using this package.

```
\usepackage{svg}
\usepackage{svg-extract}
\svgsetup{clean=true}
...
\includesvg[width=\linewidth]{foo}
```

Consider the following example image produced by **ROOT** in Figure 3. This figure was generated by the **ROOT** macro `root.C`, provided within `<texmf>/doc/latex/svg/examples/`, which produces the file `root.svg` when run. The code used to produce this SVG file from within **ROOT** is

```
void root() {

// Set the style.
gStyle->SetTextFont(80);      gStyle->SetLabelFont(80,"XYZ");
gStyle->SetTitleFont(80,"");   gStyle->SetTitleFont(80,"XYZ");
gStyle->SetPalette(1);        gStyle->SetOptStat(0);

// Draw the plot.
TH2D *h = new TH2D("", "", 25, 0, 3.9, 25, 0, 3.9); TRandom r;
for (int i = 0; i < 30000; i++) h->Fill(r.Gaus(2.,1), r.Gaus(2.,1));
h->GetXaxis()->CenterTitle(); h->GetXaxis()->SetTitleOffset(2.5);
h->GetYaxis()->CenterTitle(); h->GetYaxis()->SetTitleOffset(2.5);
h->GetXaxis()->SetTitle("\larger[2]$x$");
h->GetYaxis()->SetTitle("\larger[2]$y$");
h->Draw("LEGO2");

// Draw additional text.
TText *t = new TText(); t->SetTextAlign(31);
t->DrawText(0.7, 0.9, "\larger[2]$z(x,y) = \frac{1}{\sqrt{4\pi^2}}\exp\left(-\left(\frac{x-\mu_x}{\sigma_x}\right)^2 - \frac{2\sigma_x^2}{\sigma_y^2} + \frac{(y-\mu_y)^2}{\sigma_y^2}\right)$");

// Print the plot.
gPad->Print("root.svg");

}
```

where the text produced within the **ROOT** plot is set to a precision of zero.

The plot was then included within this document using the following \LaTeX code

```
\begin{figure}
\centering%
\includesvg[%
inkscapearea=page,height=6cm,pretex=\tiny,convertformat=png%
]{root}%
\caption{Rendering of a \app{ROOT} plot---no more \emph{Comic CERNs}}%
```

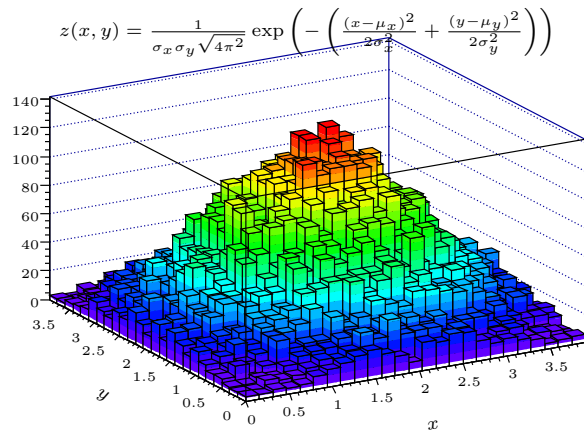


Figure 3: Rendering of a **ROOT** plot—no more *Comic CERNs*

```
\label{fig:root}%
\end{figure}
```

which includes the graphic as well as the \LaTeX file exported by **Inkscape**, produces the extracted PDF image (`root.pdf`) and converts this to a PNG image (`root.png`) by using **ImageMagick**. Enjoy plots from **ROOT** with natively rendered \LaTeX !

Part II.

Implementation

A. Initialization

A.1. Packages

The package **svg** requires **scrbase** for options processing, the packages **ifluatex**, **ifpdf** and **ifxetex** for detecting the used L^AT_EX engine, **pdftexcmds** for pdfT_EX primitives when using LuaT_EX, **shellesc** and **ifplatform** for engine independent access to systems commands and files as well as **graphicx** for the inclusion of PDF files. The usage of packages **xcolor** and **transparent** can be switched of with the corresponding options. Package **svg-extract** only needs package **svg** itself.

```
1 (*base)
2 \RequirePackage{scrbase}[2016/06/14]
3 \RequirePackage{ifpdf}[2016/05/14]
4 \RequirePackage{ifluatex}[2016/05/16]
5 \RequirePackage{ifxetex}[2010/09/12]
6 \RequirePackage{pdftexcmds}[2016/05/21]
7 \RequirePackage{shellesc}[2016/06/07]
8 \RequirePackage{trimspaces}[2009/09/17]
9 \RequirePackage{graphicx}[1999/02/16]
10 \end{base}
11 (*extract)
12 \RequirePackage{svg}[2017/03/27]
13 \end{extract}
```

A.2. Helper macros

```
\svg@tempa Internal temporary macros. The catcode for double quotes are also temporarily changed.
\svg@tempb
\svg@box
\if@svg@tempswa
14 \newcommand*\svg@tempa{}
15 \newcommand*\svg@tempb{}
16 \newbox\svg@box
17 \newif\if@svg@tempswa
18 \edef\svg@catcodecodes@restore{%
19 \catcode'\noexpand\"the\catcode'\relax%
20 }
21 \@makeother\"%
```

B. Including SVG files with package **svg**

B.1. Options

All options, which can be set either as package options or with `\svgsetup`, as well as the optional parameters for both user commands `\includesvg[parameters]{svg filename}` and `\includeinkscape[parameters]{graphic filename}` are defined with the interface provided by package **scrbase**.

```
22 \DefineFamily{SVG}
23 \DefineFamilyMember{SVG}
```


`\svg@deprecated@key` With version v2.00 the whole user interface was renewed. For reasons of compatibility, outdated options and parameters from version v1.0 are also provided. If an old key was given, a warning is issued and the valid key is used.

```

24 \newcommand*\svg@deprecated@key[3][svg]{%
25   \PackageWarning{#1}{%
26     The option key ‘#2’ is deprecated.\MessageBreak%
27     It’s recommended to use ‘#3’\MessageBreak%
28     instead%
29   }%
30   \FamilyOptions{SVG}{#3}%
31 }

```

Within the exported L^AT_EX files of *Inkscape*, some commands are used out of additional packages. But maybe the user doesn’t want to load this packages anyways.

<code>usexcolor</code> (opt.)	Options for preventing packages xcolor and transparent to be loaded.
<code>noxcolor</code> (opt.)	
<code>\if@svg@use@xcolor</code>	<code>32 \newif\if@svg@use@xcolor</code>
<code>usetransparent</code> (opt.)	<code>33 \FamilyBoolKey{SVG}{usexcolor}{@svg@use@xcolor}</code>
<code>nottransparent</code> (opt.)	<code>34 \DeclareOption{noxcolor}{\FamilyOptions{SVG}{usexcolor=false}}</code>
<code>\if@svg@use@transparent</code>	<code>35 \newif\if@svg@use@transparent</code>
	<code>36 \FamilyBoolKey{SVG}{usetransparent}{@svg@use@transparent}</code>
	<code>37 \DeclareOption{nottransparent}{\FamilyOptions{SVG}{usetransparent=false}}</code>

They are only available during the loading process of package **svg**.

```

38 \AtEndOfPackage{%
39   \RelaxFamilyKey{SVG}{usexcolor}%
40   \RelaxFamilyKey{SVG}{usetransparent}%
41   \if@svg@use@xcolor%
42     \RequirePackage{xcolor}[2016/05/11]%
43   \else%
44     \AfterPackage*{xcolor}{%
45       \PackageWarning{svg}{Package ‘xcolor’ was loaded anyway}%
46     }%
47   \fi%
48   \if@svg@use@transparent%
49     \RequirePackage{transparent}[2016/05/16]%
50   \else%
51     \AfterPackage*{transparent}{%
52       \PackageWarning{svg}{Package ‘transparent’ was loaded anyway}%
53     }%
54   \fi%
55 }

```

B.1.1. The invocation of *Inkscape*

The Application *Inkscape* is used to create includable graphic files in a desired format (PDF/EPS/PS/PNG) out of files in SVG format, whereas the support of L^AT_EX can optionally be used.

<code>inkscape</code> (opt.)	The intension of option <code>inkscape</code> is to control the running behaviour of <i>Inkscape</i> . It can
<code>\svg@ink@mode</code>	be switched off at all (<code>inkscape=false</code>) or invoked only if necessary (<code>inkscape=true</code>) or the command line call can be forced with every L ^A T _E X run (<code>inkscape=forced</code>). Additionally, option <code>inkscape</code> can be used as wrapper for options <code>inkscapeformat</code> , <code>inkscapectex</code> , <code>inkscapearea</code> and <code>inkscapecdpi</code> , which are declared later.

```

56 \newcommand*\svg@ink@mode{}
57 \DefineFamilyKey{SVG}{inkscape}[true]{%
58   \lowercase{\svg@sanitize@dq\svg@tempb{#1}}%
59   \FamilySetNumerical{SVG}{inkscape}{svg@tempa}{%
60     {false}{0},{off}{0},{no}{0},%

```

```

61 {true}{1},{on}{1},{yes}{1},{onlynewer}{1},{newer}{1},%
62 {force}{2},{forced}{2},{overwrite}{2},%
63 {pdf}{3},{eps}{4},{ps}{5},{png}{6},%
64 {drawing}{7},{crop}{7},%
65 {page}{8},{nocrop}{8},%
66 {tex}{9},{latex}{9},{exportlatex}{9},{latexexport}{9},%
67 {notex}{10},{nolatemx}{10},{noexportlatex}{10},{nolatemxexport}{10},%
68 {latexnoexport}{10},{raw}{10},{plain}{10},{simple}{10}%
69 }\svg@tempb}%
70 \ifx\FamilyKeyState\FamilyKeyStateProcessed%

```

Setting the mode for invoking *Inkscape*...

```

71 \ifnum\svg@tempa<\thr@@\relax%
72 \let\svg@ink@mode\svg@tempa%
73 \else%

```

...and the part as wrapper for different options.

```

74 \ifcase\svg@tempa\relax\or\or\or% pdf
75 \FamilyOptions{SVG}{inkscapeformat=pdf}%
76 \or% eps
77 \FamilyOptions{SVG}{inkscapeformat=eps}%
78 \or% ps
79 \FamilyOptions{SVG}{inkscapeformat=ps}%
80 \or% png
81 \FamilyOptions{SVG}{inkscapeformat=png}%
82 \or% drawing
83 \FamilyOptions{SVG}{inkscapearea=drawing}%
84 \or% page
85 \FamilyOptions{SVG}{inkscapearea=page}%
86 \or% tex
87 \FamilyOptions{SVG}{inkscapelatex=true}%
88 \or% notex
89 \FamilyOptions{SVG}{inkscapelatex=false}%
90 \fi%
91 \fi%

```

It's also possible to set the option `inkscape dpi` by passing a number followed by `dpi` like `inkscape=300dpi`.

```

92 \else% dpi
93 \def\svg@tempa##1dpi##2\@nil{%
94 \ifstr{##2}{dpi}{\FamilyOptions{SVG}{inkscape dpi=##1}}{%
95 }%
96 \lowercase{\expandafter\svg@tempa\svg@tempb dpi \@nil}%

```

In version v1.0 the option `inkscape` was used to set both the executable and options for *Inkscape*. This is taken into account here.

```

97 \ifx\FamilyKeyState\FamilyKeyStateProcessed\else%

```

Splitting executable from options with delimited macros. After calling `\svg@tempa` with the given value, the part for the executable is stored in `\svg@tempa` and the option part—which is recognized by the first `-` character—in `\svg@tempb`.

```

98 \svg@quotes@remove[#{1}]{\svg@tempb}%
99 \def\svg@tempa##1-##2\@nil{%
100 \IfArgIsEmpty{##2}{\def\svg@tempb{} }{%
101 \def\svg@tempa####1-\@nil{\def\svg@tempb{-####1}}%
102 \svg@tempa##2\@nil%
103 }%
104 \edef\svg@tempa{\trim@spaces{##1}}%
105 }%
106 \edef\svg@tempb{%
107 \noexpand\svg@tempa\svg@tempb-\noexpand\@nil%
108 }%

```

```

109 \svg@tempb%
110 \if@svg@quotes@found%
111 \edef\svg@tempa{"\svg@tempa"}%
112 \fi%
113 \PackageWarning{svg}{%
114 Setting the executable%
115 \ifx\svg@tempb\@empty\else%
116 \space and associated options%
117 \fi%
118 \MessageBreak%
119 for Inkscape should be done with options\MessageBreak%
120 'inkscapeexe=\svg@tempa'%
121 \ifx\svg@tempb\@empty\else%
122 \MessageBreak and 'inkscapeopt=\svg@tempb'%
123 \fi.\MessageBreak%
124 Nevertheless, this was done by now anyway%
125 }%
126 \edef\svg@tempa{%
127 \noexpand\FamilyOptions{SVG}{inkscapeexe=\svg@tempa}%
128 \ifx\svg@tempb\@empty\else%
129 \noexpand\FamilyOptions{SVG}{inkscapeopt=\svg@tempb}%
130 \fi%
131 }%
132 \svg@tempa%
133 \fi%
134 \fi%
135 }

```

on (opt.) Package options which can be used to switch functionality on or off during the loading of
off (opt.) package **svg**.

```

136 \DeclareOption{on}{\FamilyOptions{SVG}{inkscape=true}}
137 \DeclareOption{off}{\FamilyOptions{SVG}{inkscape=false}}

```

inkscapeformat (opt.) With option `inkscapeformat` the output format of the **Inkscape** export function, which
\svg@ink@format is called via `\ShellEscape`, can be configured. It is set to pdf or, if dvi output could be
detected, to eps during initialization.

```

138 \newcommand*\svg@ink@format{pdf}
139 \ifxetex\else\ifpdf\else
140 \renewcommand*\svg@ink@format{eps}
141 \fi\fi
142 \DefineFamilyKey{SVG}{inkscapeformat}{%
143 \lowercase{\def\svg@tempa{#1}}%
144 \FamilySetNumerical{SVG}{inkscapeformat}{\svg@tempa}{%
145 {pdf}{0},{eps}{1},{ps}{2},{png}{3}%
146 }{\svg@tempa}%
147 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
148 \ifcase\svg@tempa\relax% latex
149 \renewcommand*\svg@ink@format{pdf}%
150 \or% eps
151 \renewcommand*\svg@ink@format{eps}%
152 \or% ps
153 \renewcommand*\svg@ink@format{ps}%
154 \or% png
155 \renewcommand*\svg@ink@format{png}%
156 \fi%
157 \fi%
158 }

```

inksca Patelatex (opt.) This option controls whether the **Inkscape** export will be invoked with or without the
latex (opt.) generation of a separate L^AT_EX file.

```

\svg@ink@latex tex (opt.)
159 \newif\if@svg@ink@latex
160 \FamilyBoolKey{SVG}{inksca Patelatex}{@svg@ink@latex}

```

```
161 \FamilyBoolKey{SVG}{\latex}{@svg@ink@latex}
162 \FamilyBoolKey{SVG}{\tex}{@svg@ink@latex}
```

`inkscapearea` (opt.) The exported area for an *Inkscape* graphic can be set with this option.

```
\svg@ink@area
163 \newcommand*\svg@ink@area{}
164 \DefineFamilyKey{SVG}{inkscapearea}{%
165   \FamilySetNumerical{SVG}{inkscapearea}{svg@tempa}{%
166     {drawing}{0},{crop}{0},%
167     {page}{1},{nocrop}{1}%
168   }{#1}%
169   \ifx\FamilyKeyState\FamilyKeyStateProcessed%
170     \ifcase\svg@tempa\relax% drawing
171       \renewcommand*\svg@ink@area{-D}%
172     \else% page
173       \renewcommand*\svg@ink@area{-C}%
174     \fi%
175   \fi%
176 }
```

`inkscapedpi` (opt.) A density can be chosen, which is used during export with *Inkscape* for bitmaps and
`inkscapedensity` (opt.) rasterization of filters.

```
\svg@ink@dpi
177 \newcommand*\svg@ink@dpi{}
178 \let\svg@ink@dpi\relax
179 \DefineFamilyKey{SVG}{inkscapedpi}{%
180   \FamilyKeyStateUnknownValue%
181   \svg@ifvalueisrelax{#1}{%
182     \let\svg@ink@dpi\relax%
183     \FamilyKeyStateProcessed%
184   }{%
185     \def\svg@tempa##1dpi##2\@nil{\def\svg@tempa{##1}}%
186     \lowercase{\svg@tempa#1dpi\@nil}%
187     \ifnumber{\svg@tempa}{%
188       \edef\svg@ink@dpi{\svg@tempa}%
189       \FamilyKeyStateProcessed%
190     }{}}%
191   }%
192 }
193 \DefineFamilyKey{SVG}{inkscapedensity}{\FamilyOptions{SVG}{inkscapedpi=#1}}
```

`inkscapeexe` (opt.) With these options, the terminal command for invoking *Inkscape* as well as additional
`\svg@ink@exe` options can be defined.

```
\svg@ink@opt
194 \newcommand*\svg@ink@exe{inkscape}
195 \DefineFamilyKey{SVG}{inkscapeexe}{%
196   \renewcommand*\svg@ink@exe{#1}%
197   \FamilyKeyStateProcessed%
198 }
199 \newcommand*\svg@ink@opt{}
200 \DefineFamilyKey{SVG}{inkscapeopt}{%
201   \renewcommand*\svg@ink@opt{#1}%
202   \FamilyKeyStateProcessed%
203 }
```

B.1.2. Setting input folder and file

`svgpath` (opt.) In version v1.0 setting the path to SVG files was done via option. So this method is provided as well.

```
204 \DefineFamilyKey{SVG}{svgpath}{%
205   \PackageWarning{svg}{%
206     The key ‘svgpath’ is deprecated. It’s recommended\MessageBreak%
207     to use ‘\string\svgpath’ instead%
```

```

208 }%
209 \ifx\svgpath\@undefined%
210 \AtEndOfPackage{\svgpath{#{1}}}%
211 \else%
212 \svgpath{#{1}}%
213 \fi%
214 \FamilyKeyStateProcessed%
215 }

```

`svgextension` (opt.) This option modifies the expected extension for the input file which is exported with *Inkscape*. It is set to `svg` by default.

```

extension (opt.)
  ext (opt.)
  \svg@file@ext
216 \newcommand*\svg@file@ext{svg}
217 \DefineFamilyKey{SVG}{svgextension}{%

```

The extension should be in lower case letters.

```

218 \lowercase{\svg@quotes@remove[#{1}]{\svg@file@ext}}%

```

Remove leading dots from the extension.

```

219 \svg@remove@leadingchar.\svg@file@ext%
220 }
221 \DefineFamilyKey{SVG}{extension}{\FamilyOptions{SVG}{svgextension=#1}}
222 \DefineFamilyKey{SVG}{ext}{\FamilyOptions{SVG}{svgextension=#1}}

```

B.1.3. Setting output folder

`inkscapepath` (opt.) The option `inkscapepath` controls, in which folder the results of the *Inkscape* export will be located. With option `inkscapepath` the name of the exported file itself can be changed.

```

inkscapepath (opt.)
inkscapepath (opt.)
  \svg@out@path
  \svg@out@name
  \svg@out@base
223 \newcommand*\svg@out@path{}
224 \newcommand*\svg@out@name{\svg@file@name\svg@file@suffix}
225 \newcommand*\svg@out@base{\svg@out@path\svg@out@name.\svg@ink@format}
226 \DefineFamilyKey{SVG}{inkscapepath}{%
227 \svg@sanitize@dq\svg@tempb{#{1}}%
228 \FamilySetNumerical{SVG}{inkscapepath}{\svg@tempa}{%
229 {svgpath}{0},{svgdir}{0},%
230 {svgsubpath}{1},{svgsubdir}{1},%
231 {basepath}{2},{basedir}{2},{jobpath}{2},{jobdir}{2},%
232 {basesubpath}{3},{basesubdir}{3},{jobsubpath}{3},{jobsubdir}{3}%
233 }\svg@tempb}%
234 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
235 \ifcase\svg@tempa\relax% svgpath
236 \renewcommand*\svg@out@path{\svg@file@path}%
237 \or% svgsubpath
238 \renewcommand*\svg@out@path{\svg@file@path\svg-inkscape/}%
239 \or% basepath
240 \renewcommand*\svg@out@path{./}%
241 \or% basesubpath
242 \renewcommand*\svg@out@path{./svg-inkscape/}%
243 \fi%
244 \else%
245 \edef\svg@out@path{\svg@tempb}%
246 \svg@normalize@path{\svg@out@path}%
247 \FamilyKeyStateProcessed%
248 \fi%
249 }
250 \DefineFamilyKey{SVG}{inkscapepath}{%
251 \renewcommand*\svg@out@name{#{1}\svg@file@suffix}%
252 \FamilyKeyStateProcessed%
253 }

```

B.1.4. Options for the inclusion of graphics

After the graphic export with *Inkscape*, the inclusion of those graphics can be controlled with the following options.

<pre> width (opt.) \svg@param@width height (opt.) \svg@param@width distort (opt.) keepaspectratio (opt.) \if@svg@param@distort scale (opt.) \svg@param@scale </pre>	<pre> These options determine the size of the included graphics. The usage of \relax as value resets the respective option to the default behavior. 254 \newcommand*\svg@param@width{\z@} 255 \DefineFamilyKey{SVG}{width}{% 256 \FamilyKeyStateUnknownValue% 257 \svg@ifvalueisrelax{#1}{% 258 \renewcommand*\svg@param@width{\z@}% 259 \FamilyKeyStateProcessed% 260 }{% 261 \FamilySetLengthMacro{SVG}{width}{\svg@param@width}{#1}% 262 \ifx\FamilyKeyState\FamilyKeyStateProcessed% 263 \ifdim\svg@param@width<\z@\relax% 264 \FamilyKeyStateUnknownValue% 265 \fi% 266 \fi% 267 }% 268 } 269 \newcommand*\svg@param@height{\z@} 270 \DefineFamilyKey{SVG}{height}{% 271 \FamilyKeyStateUnknownValue% 272 \svg@ifvalueisrelax{#1}{% 273 \renewcommand*\svg@param@height{\z@}% 274 \FamilyKeyStateProcessed% 275 }{% 276 \FamilySetLengthMacro{SVG}{height}{\svg@param@height}{#1}% 277 \ifx\FamilyKeyState\FamilyKeyStateProcessed% 278 \ifdim\svg@param@height<\z@\relax% 279 \FamilyKeyStateUnknownValue% 280 \fi% 281 \fi% 282 }% 283 } 284 \newif\if@svg@param@distort 285 \FamilyBoolKey{SVG}{distort}{@svg@param@distort} 286 \DefineFamilyKey{SVG}{keepaspectratio}[true]{% 287 \FamilySetBool{SVG}{keepaspectratio}{@svg@tempswa}{#1}% 288 \ifx\FamilyKeyState\FamilyKeyStateProcessed% 289 \if@svg@tempswa% 290 \FamilyOptions{SVG}{distort=false}% 291 \else 292 \FamilyOptions{SVG}{distort=true}% 293 \fi% 294 \fi% 295 } 296 \newcommand*\svg@param@scale{1} 297 \DefineFamilyKey{SVG}{scale}{% 298 \FamilyKeyStateUnknownValue% 299 \svg@ifvalueisrelax{#1}{% 300 \renewcommand*\svg@param@scale{1}% 301 \FamilyKeyStateProcessed% 302 }{% 303 \ifisdimension{#1\p}{% 304 \ifdim\dimexpr#1\p@\relax>\z@\relax% 305 \renewcommand*\svg@param@scale{#1}% 306 \FamilyKeyStateProcessed% 307 \fi% 308 }{% 309 }% 310 } </pre>
---	--

`pretex` (opt.) For executing code right before or after the graphic inclusion, two hooks are defined.

```

\svg@param@pretex
  apptex (opt.)
\svg@param@apptex
  postex (opt.)
311 \newcommand*\svg@param@pretex{}
312 \let\svg@param@pretex\relax
313 \DefineFamilyKey{SVG}{pretex}{%
314   \svg@ifvalueisrelax{#1}{%
315     \let\svg@param@pretex\relax%
316   }{%
317     \def\svg@param@pretex{#1}%
318   }%
319   \FamilyKeyStateProcessed%
320 }
321 \newcommand*\svg@param@apptex{}
322 \let\svg@param@apptex\relax
323 \DefineFamilyKey{SVG}{apptex}{%
324   \svg@ifvalueisrelax{#1}{%
325     \let\svg@param@apptex\relax%
326   }{%
327     \def\svg@param@apptex{#1}%
328   }%
329   \FamilyKeyStateProcessed%
330 }
331 \DefineFamilyKey{SVG}{postex}{%
332   \svg@deprecated@key{postex=#1}{apptex=#1}%
333 }

```

`lastpage` (opt.) For *Inkscape* 0.91 a bug concerning the L^AT_EX export has been reported (<https://bugs.launchpad.net/ubuntu/+source/inkscape/+bug/1417470>). Sometimes the L^AT_EX file created by *Inkscape* tries to include more pages than actually are present in the PDF file. To work around this problem, a patch is provided. For this purpose, the total page number is read from the PDF file.

```

\svg@param@lastpage (counter)
334 \newcounter{svg@param@lastpage}
335 \DefineFamilyKey{SVG}{lastpage}{%
336   \FamilySetNumerical{SVG}{lastpage}{svg@tempa}{%
337     {false}{0},{off}{0},{no}{0},{ignore}{0},%
338     {true}{1},{on}{1},{yes}{1},{auto}{1}%
339   }{#1}%
340   \ifx\FamilyKeyState\FamilyKeyStateProcessed%
341     \ifcase\svg@tempa\relax% false
342     \FamilySetCounter{SVG}{lastpage}{svg@param@lastpage}{\m@ne}%
343     \or% true
344     \FamilySetCounter{SVG}{lastpage}{svg@param@lastpage}{\z@}%
345     \fi%
346   \fi%
347 }

```

`draft` (opt.) The option `draft` has the same effect as the eponymous option of package **graphicx**.

```

\if@svg@draft
348 \newif\if@svg@draft
349 \FamilyBoolKey{SVG}{draft}{@svg@draft}
350 \AtBeginDocument{\if@svg@draft\else\ifGin@draft\@svg@drafttrue\fi\fi}

```

B.2. Handling path information

Both packages **svg** and **svg-extract** should be able to handle user-defined input and output paths. As there is the possibility for users to provide paths with or without quotes to L^AT_EX, this is taken into account.

`\svg@deactivate@dq` In order to avoid errors concerning file names with package **babel** and its active double quotes, this command is defined.

```

351 \newcommand*\svg@deactivate@dq{}

```

```

352 \AfterPackage+{babel}{%
353   \renewcommand*{svg@deactivate@dq}{\bbl@deactivate{}}%
354   \providecommand*\bbl@deactivate[1]{}%
355 }

```

`\svg@sanitize@dq` Save expansion of the second argument in the macro from the first argument with deactivated double quotes.

```

356 \newcommand*{svg@sanitize@dq}[2]{%
357   \begingroup%
358   \svg@deactivate@dq%
359   \edef\svg@tempa{\endgroup\def\noexpand#1{#2}}%
360   \svg@tempa%
361 }

```

`\svg@quotes@remove` and `\svg@quotes@@remove` These two commands are used to remove all occurring quotes within a string. The only argument passed to `\svg@quotes@remove` is not the string itself but a macro in which a string is stored.

```

362 \newcommand*{svg@quotes@remove}[2][ ]{%
363   \begingroup%
364   \IfArgIsEmpty{#1}{\def\svg@tempb{#2}}{\def\svg@tempb{#1}}%
365   \svg@sanitize@dq\svg@tempa{\svg@tempb}%
366   \expandafter\svg@quotes@check\expandafter{\svg@tempa}%
367   \expandafter\svg@quotes@@remove\svg@tempa""\nil%
368   \edef\svg@tempb{%
369     \endgroup%
370     \def\noexpand#2{\svg@tempa}%
371     \if@svg@quotes@found%
372       \noexpand\@svg@quotes@foundtrue%
373     \else%
374       \noexpand\@svg@quotes@foundfalse%
375     \fi%
376   }%
377   \svg@tempb%
378 }
379 \newcommand*{svg@quotes@@remove}{%
380 \def\svg@quotes@@remove#1"#2"#3\nil{%
381   \IfArgIsEmpty{#2}{%
382     \edef\svg@tempa{#1}%
383   }{%
384     \svg@quotes@@remove#1#2#3""\nil%
385   }%
386 }

```

`\svg@quotes@check` and `\if@svg@quotes@found` During the treatment of paths, it may be necessary to temporarily remove quotes and, if required, add them again later. For this purpose, the switch `\if@svg@quotes@found` as well as the commands `\svg@quotes@check` and `\svg@quotes@@check`, which controls the switch, are defined. As before, the string is passed in a macro to `\svg@quotes@check`.

```

387 \newif\if@svg@quotes@found
388 \newcommand*{svg@quotes@check}[1]{%
389   \expandafter\svg@quotes@@check#1""\nil%
390 }
391 \newcommand*{svg@quotes@@check}{%
392 \def\svg@quotes@@check#1"#2\nil{%
393   \IfArgIsEmpty{#2}{\@svg@quotes@foundfalse}{\@svg@quotes@foundtrue}%
394 }

```

`\svg@remove@leadingchar` This command removes the single character in given with the first argument from the expanded macro in the second argument.

```

395 \newcommand*{svg@remove@leadingchar}[2]{%
396   \begingroup%

```



```

397 \svg@sanitize@dq\svg@tempa{#2}%
398 \def\svg@tempb{%
399 \def\svg@tempa###1\@nil{\def\svg@tempa{###1}}%
400 \kernel@ifnextchar#1%
401 {\expandafter\svg@tempa\@gobble}%
402 {\svg@tempa}%
403 }%
404 \expandafter\svg@tempb\svg@tempa\@nil%
405 \edef\svg@tempb{%
406 \endgroup%
407 \def\noexpand#2{\svg@tempa}%
408 }%
409 \svg@tempb%
410 }

```

`\svg@set@input@path` In order to import SVG files from different folders, `\svg@set@input@path` evaluates several macros, which are supposed to be used for holding different search folders. Any given path will be handled by `\svg@normalize@path`. The optional argument can be used to append an additional search path.

```

411 \newcommand*\svg@set@input@path[1] [] {%
412 \begingroup%
413 \svg@deactivate@dq%

```

If a path was already found and stored within `\svg@file@path`, it is searched first and wrapped in curly braces. This is necessary for using commands like `\input{(tex filename)}` within SVG files.

```

414 \ifx\svg@file@path\@empty\else%
415 \svg@normalize@path{\svg@file@path}%
416 \edef\svg@file@path{\{\svg@file@path\}}%
417 \fi%

```

Afterwards, several search paths are appended. If `\svgpath` was used, it is searched next. If nothing was found, `\graphicspath` is considered if defined followed by a path given in the third argument. If nothing was found yet, the standard `\input@path` is searched last.

```

418 \svg@append@input@path{\svg@file@path}\{\svg@input@path}%
419 \svg@append@input@path{\svg@file@path}\{\Ginput@path}%
420 \IfArgIsEmpty{#1}\{\{\svg@append@input@path{\svg@file@path}\{#1}\}}%
421 \svg@append@input@path{\svg@file@path}\{\input@path}%

```

Finally, `\input@path` is set.

```

422 \edef\svg@tempa{%
423 \endgroup%
424 \ifx\svg@file@path\@empty\else%
425 \def\noexpand\input@path{\svg@file@path}%
426 \fi%
427 }%
428 \svg@tempa%
429 }

```

Only, if a certain search path is defined, it is added. The paths given in the first argument are compared to each path in the second argument and only new ones are added.

```

430 \newcommand*\svg@append@input@path[2] {%
431 \ifx#2\@undefined\else%
432 \edef\svg@tempb{#2}%
433 \expandafter\@tfor\expandafter\svg@tempa\expandafter:\expandafter=%
434 \svg@tempb\do{%

```

Passing each new path to `\svg@normalize@path`. If a path already exists, switch `\if@svg@tempswa` is set to false.

```

435     \ifx\svg@tempa\@empty\else%
436     \@svg@tempswatrue%
437     \svg@normalize@path{\svg@tempa}%
438     \expandafter\@tfor\expandafter\svg@tempb\expandafter:\expandafter=%
439     #1\do{%
440     \ifx\svg@tempa\svg@tempb%
441     \@svg@tempswafalse%
442     \@break@tfor%
443     \fi%
444     }%
445     \if@svg@tempswa%
446     \edef#1{#1\svg@tempa}%
447     \fi%
448     \fi%
449     }%
450 \fi%
451 }

```

`\svg@normalize@path` If any path is given, a trailing slash is needed. These two macros ensure that this condition is fulfilled in any case, even if this is not considered by the user. As before, a macro containing the path string is passed to `\svg@normalize@path`.

`\svg@normalize@@path`

```

452 \newcommand*\svg@normalize@path[1]{%
453 \begingroup%
454 \svg@quotes@remove[#{1}]{\svg@tempa}%
455 \ifx\svg@tempa\@empty\relax%
456 \def\svg@tempa{./}%
457 \fi%
458 \expandafter\svg@normalize@@path\svg@tempa//\@nil%
459 \edef\svg@tempb{%
460 \endgroup%
461 \if@svg@quotes@found%
462 \def\noexpand#1{"\svg@tempa"%
463 \else%
464 \def\noexpand#1{\svg@tempa}%
465 \fi%
466 }%
467 \svg@tempb%
468 }
469 \newcommand*\svg@normalize@@path{}
470 \def\svg@normalize@@path#1/#2/\@nil{%
471 \IfArgIsEmpty{#2}{%
472 \IfArgIsEmpty{#1}{\def\svg@tempa{}}{\def\svg@tempa{#1/}}%
473 }{%
474 \svg@normalize@@path#2/\@nil%
475 \edef\svg@tempa{#1/\svg@tempa}%
476 }%
477 }

```

`\svg@ifvalueisrelax` For some keys the usage of `\relax` as a value should lead to a special reaction, such as restoring to default behavior or resetting the key. Therefore, `\svg@ifvalueisrelax` checks, whether `\relax` was used as value or not.

```

478 \newcommand*\svg@ifvalueisrelax[1]{%
479 \begingroup%
480 \def\svg@tempa{#1}%
481 \def\svg@tempb{\relax}%
482 \ifx\svg@tempa\svg@tempb\relax%
483 \aftergroup\@firstoftwo%
484 \else%
485 \aftergroup\@secondoftwo%
486 \fi%

```

```
487 \endgroup%
488 }
```

`\svg@get@path` The command `\svg@get@path` tries to find a given SVG file. If the searched file wasn't found in the current path, all paths given with `\svg@path` are evaluated. If there was no appropriate file again, all paths given by `\graphicspath` are examined. In the last step, a given path within the second mandatory argument is browsed. The results for file path and name are stored in `\svg@file@path` and `\svg@file@name` as well as the compound of both is saved in `\svg@file@base`.

```
489 \newif\if@svg@file@found
490 \newcommand*\svg@file@path{}
491 \newcommand*\svg@file@name{}
492 \newcommand*\svg@file@base{}
493 \newcommand*\svg@file@suffix{}
494 \newcommand*\svg@get@path[3][\svg@file@ext]{%
495 \begingroup%
```

A maybe given, unneeded file extension is removed.

```
496 \svg@filename@parse[#{1}]{#2}%
497 \IfArgIsEmpty{#1}{%
498 \edef\svg@tempa{\filename@area\filename@base.\filename@ext}%
499 }{%
500 \edef\svg@tempa{\filename@area\filename@base.#1}%
501 }%
```

After calling `\svg@set@input@path`, all search paths are stored in `\input@path`, a single path given in the third argument will also be considered.

```
502 \svg@set@input@path[#{3}]%
```

The specified file is searched with `\IfFileExists`. If the file search was successful, the macro `\svg@filename@parse` is called with the result.

```
503 \@svg@tempswafalse%
504 \expandafter\IfFileExists\expandafter{\svg@tempa}{%
505 \expandafter\svg@quotes@check\expandafter{\svg@tempa}%
506 \if@svg@quotes@found\else%
507 \svg@quotes@remove{\@file@und}%
508 \fi%
509 \@svg@tempswatruetrue%
510 \edef\@file@und{\expandafter\trim@spaces\expandafter{\@file@und}}%
511 \svg@filename@parse[#{1}]{\@file@und}%
512 }}%
513 \edef\svg@tempa{%
514 \endgroup%
515 \if@svg@tempswa%
516 \noexpand\@svg@file@foundtrue%
517 \def\noexpand\svg@file@path{\filename@area}%
518 \def\noexpand\svg@file@name{\filename@base}%
519 \def\noexpand\svg@file@base{\filename@area\filename@base}%
520 \else%
521 \noexpand\@svg@file@foundfalse%
522 \def\noexpand\svg@file@path{}%
523 \def\noexpand\svg@file@name{#2}%
524 \def\noexpand\svg@file@base{#2}%
525 \fi%
526 }%
527 \svg@tempa%
528 }
```

`\svg@filename@parse` As the internal L^AT_EX 2_ε command `\filename@parse` is not able to split a given file name containing quotes, `\svg@filename@parse` is defined to resolve this problem. The optional

argument can be used to give a specific file extension, which should be searched within `\filename@ext`. If found at the very end, the previous part is appended to `\filename@base`.

```
529 \newcommand*{svg@filename@parse}[2][]{%
530   \begingroup%
```

The given path and file is parsed with `\filename@parse`.

```
531   \svg@sanitize@dq\svg@tempa{#2}%
532   \expandafter\filename@parse\expandafter{\svg@tempa}%
533 % If there are quotes in the file path, the closing one will be found as first
534 % character in \cs{filename@base} as \cs{filename@area} is splitted at the last
535 % slash. This leading quote is removed from \cs{filename@base} with
536 % \cs{svg@remove@leadingchar}.
537 %   \begin{macrocode}
538   \svg@quotes@remove{\filename@area}%
539   \if@svg@quotes@found%
540     \edef\filename@area{"\filename@area"%
541     \svg@remove@leadingchar"\filename@base}%
542   \fi%
```

The found extension is parsed against the optional argument. If a double quote was found within the extension, it actually belongs to `\filename@base`.

```
543   \ifx\filename@ext\relax\else%
544     \svg@quotes@remove{\filename@ext}%
545     \svg@extension@parse{#1}%
546     \if@svg@quotes@found%
547       \edef\filename@base{\filename@base"%
548       \fi%
549   \fi%
```

Quotes within `\filename@base` are normalized.

```
550   \svg@quotes@remove{\filename@base}%
551   \if@svg@quotes@found%
552     \edef\filename@base{"\filename@base"%
553   \fi%
```

With `\svg@tempa` the group is closed and the results are saved in the macros `\filename@...`

```
554   \edef\svg@tempa{%
555     \endgroup%
556     \def\noexpand\filename@area{\filename@area}%
557     \def\noexpand\filename@base{\filename@base}%
558     \ifx\filename@ext\relax%
559       \let\noexpand\filename@ext\noexpand\relax%
560     \else%
561       \def\noexpand\filename@ext{\filename@ext}%
562     \fi%
563   }%
564   \svg@tempa%
565 }
```

`\svg@extension@parse`
`\svg@extension@@parse`

These macros are used to permit multiple dots in file names. The content of `\filename@ext` is split at each occurrence of `.` and the trailing part is compared against the content of the argument of `\svg@extension@parse`, which is probably `\svg@file@ext`. If they are equal, the previous part is appended to `\filename@base` and `\filename@ext` is set to the content of the first argument.

```
566 \newcommand*{svg@extension@parse}[1]{%
567   \IfArgIsEmpty{#1}{-}{%
568     \@expandtwoargs\ifstr%
569     {\detokenize\expandafter{\filename@ext}}{\detokenize\expandafter{#1}}{-}{%
570   \begingroup%
```

Macro `\svg@tempa` is used to temporarily store anything before the searched extension at the end of `\filename@ext` and `\svg@tempb` is set to the actual searched extension if found.

```

571     \edef\svg@tempa{%
572         \def\noexpand\svg@tempa{}}%
573     \let\noexpand\svg@tempb\relax%
574     \noexpand\svg@extension@@parse%
575     \filename@ext.\noexpand\@nil#1\noexpand\@nil%
576     }%
577     \svg@tempa%
578     \edef\svg@tempa{%
579         \endgroup%
```

If the trailing extension was found, `\filename@base` and `\filename@ext` are adopted.

```

580         \def\noexpand\filename@base{\filename@base\svg@tempa}%
581         \ifx\svg@tempb\relax%
582             \let\noexpand\filename@ext\relax%
583         \else%
584             \def\noexpand\filename@ext{\svg@tempb}%
585         \fi%
586     }%
587     \svg@tempa%
588 }%
589 }%
590 }
```

Macro `\svg@extension@@parse` is recursively called as long as there are any dots or the searched extension is found.

```

591 \newcommand*\svg@extension@@parse{}
592 \def\svg@extension@@parse#1.#2\@nil#3\@nil{%
593     \edef\svg@tempa{\svg@tempa.#1}%
594     \IfArgIsEmpty{#2}{-}{%
595         \ifstr{\detokenize{#2}}{\detokenize{#3.}}{-%
```

If the trailing extension is found, `\svg@tempb` is defined.

```

596     \edef\svg@tempb{#3}%
597     }{%
598     \svg@extension@@parse#2\@nil#3\@nil%
599     }%
600 }%
601 }
```

`\svg@file@missing` The error message, which is raised, if a file is missing either after the export with *Inkscape* or in general.

```

602 \newcommand*\svg@file@missing[3] []{%
603     \begingroup%
604     \svg@quotes@remove[#{2}]{\svg@tempa}%
605     \svg@filename@parse[#{1}]{\svg@tempa}%
606     \IfArgIsEmpty{#1}{-%
607         \svg@quotes@remove[#{3}]{\svg@tempb}%
608         \def\svg@tempa{%
609             Did you run the export with Inkscape? There's no file\MessageBreak%
610             '\filename@area\filename@base.\filename@ext'\MessageBreak%
611             although '\svg@tempb' was found.%
612         }%
613     }{%
614         \edef\filename@ext{#1}%
615         \ifstr{\filename@area}{./}{\let\filename@area\@empty}{-%
```

Collecting all considered path for the error message.

```

616     \edef\svg@tempb{#3}%
617     \ifstrf\svg@tempb}{. /}{\let\svg@tempb\@empty}{}%
618     \ifx\svg@tempb\@empty%
619         \svg@set@input@path%
620     \else%
621         \svg@set@input@path[\svg@tempb]%
622     \fi%
623     \ifx\input@path\@undefined%
624         \def\svg@tempb{No additional path was given.}%
625     \else%
626         \def\svg@tempb{Following folders have additionally been searched:}%
627         \expandafter\@tfor\expandafter\svg@tempa\expandafter:\expandafter=%
628             \input@path\do{%
629             \edef\svg@tempb{\svg@tempb\noexpand\MessageBreak\svg@tempa}%
630             }%
631     \fi%

```

The error message itself.

```

632     \def\svg@tempa{%
633         There's no file '\filename@base.\filename@ext'\MessageBreak%
634         \ifx\filename@area\@empty%
635             neither in the current directory nor any other searched\MessageBreak%
636             path given by \string\svgpath\space or \string\graphicspath.%
637             \MessageBreak\svg@tempb%
638         \else%
639             in folder '\filename@area'.%
640         \fi%
641     }%
642 }%
643 \PackageError{svg}{%
644     File '\filename@base.\filename@ext' is missing%
645 }{\svg@tempa}%
646 \endgroup%
647 }

```

`\svg@iffilenewer` The macro `\svg@iffilenewer` is used to decide, whether the export with *Inkscape* is necessary due to an updated SVG file. This can only be done, if `\pdf@filemoddate` or `\filemoddate` is defined.

```

648 \newcommand*\svg@iffilenewer[2]{\@gobbletwo}
649 \ifx\pdf@filemoddate\undefined
650 \ifx\filemoddate\undefined\else
651 \ifx\strcmp\undefined\else
652 \renewcommand*\svg@iffilenewer[2]{%
653     \begingroup%
654         \edef\svg@tempa{\filemoddate{#1}}%
655         \edef\svg@tempb{\filemoddate{#2}}%
656         \ifnum\strcmp{\svg@tempa}{\svg@tempb}>\z@\relax%
657             \aftergroup\@firstoftwo%
658         \else%
659             \aftergroup\@secondoftwo%
660         \fi%
661     \endgroup%
662 }%
663 \fi
664 \fi
665 \else
666 \ifx\pdf@strcmp\undefined\else
667 \renewcommand*\svg@iffilenewer[2]{%
668     \begingroup%
669         \edef\svg@tempa{\pdf@filemoddate{#1}}%
670         \edef\svg@tempb{\pdf@filemoddate{#2}}%
671         \ifnum\pdf@strcmp{\svg@tempa}{\svg@tempb}>\z@\relax%

```

```

672         \aftergroup\@firstoftwo%
673         \else%
674         \aftergroup\@secondoftwo%
675         \fi%
676     \endgroup%
677 }%
678 \fi
679 \fi

```

B.3. Optional Parameters for user commands

`\svg@local@param@set` Most of the package options can also be used as optional parameters for `\includesvg` or `\includeinkscape`. Some of them are overloaded for the usage as optional argument and there are some keys, which *only* can be used as optional parameters. This is realized in such a way that `\svg@local@param@use` is extended with `\svg@local@param@def` by the definition of local keys during the loading of package **svg**.

```

680 \newcommand*\svg@local@param@set[1]{%
681   \svg@local@param@use%
682   \FamilyOptions{SVG}{#1}%

```

As `\svg@local@param@set` is always used in a local group, it is possible to set `inkscapelatex` to `false`, if the output format was set to `png` with option `inkscapeformat`.

```

683   \ifstr{\svg@ink@format}{png}{\FamilyOptions{SVG}{inkscapelatex=false}}{}%

```

Using `distort=true` is only reasonable, if `height` and `width` are given.

```

684   \@svg@tempswafalse%
685   \ifdim\svg@param@width>\z\relax\ifdim\svg@param@height>\z\relax%
686     \@svg@tempswatruetrue%
687   \fi\fi%
688   \if@svg@tempswa\else%
689     \FamilyOptions{SVG}{distort=false}%
690   \fi%
691 }
692 \newcommand*\svg@local@param@use{}
693 \newcommand*\svg@local@param@def[1]{%
694   \edef\svg@local@param@use{%
695     \unexpanded\expandafter{\svg@local@param@use}\unexpanded{#1}%
696   }%
697 }

```

The family member is defined for both **svg** and **svg-extract**.

```

698 <*body>
699 \DefineFamilyMember[.param]{SVG}
700 </body>

```

B.4. User commands

`\svgsetup` The macro `\svgsetup` can be used to change options after loading the package **svg** both in preamble and the document body. For compatibility reasons, `\setsvg` is also defined.

```

701 \newcommand*\svgsetup{\FamilyOptions{SVG}}
702 \newcommand*\setsvg{\FamilyOptions{SVG}}

```

`\svgpath` With `\svgpath` the user can give several root paths to SVG files in the same way as `\graphicspath` is used. The only difference is that a missing slash is added at the end of the path, if needed.

```

703 \newcommand*\svg@input@path{}
704 \let\svg@input@path\input@path

```

```

705 \newcommand*{svgpath[1]}{%
706   \def\svg@tempa##1\@nil{%
707     \ifx\svg@tempb\bgroup%
708       \def\svg@input@path{#1}%
709     \else%
710       \def\svg@input@path{{#1}}%
711     \fi%
712   }%
713   \futurelet\svg@tempb\svg@tempa#1\@nil%
714 }

```

`\includesvg` For the inclusion of SVG files the command `\includesvg` is defined.

```

715 \newcommand*{includesvg[2] []}{%
716   \begingroup%

```

Checking for deprecated commands `\svgwidth` and `\svgscale`.

```

717   \svg@deprecated@param%

```

`inkscape` (param.) Most of the optional parameters have the same effect as the identically named options.
`inkscapeformat` (param.) Only parameter `lastpage` is extended (see below). Moreover, there are some additional
`inkscapelatex` (param.) parameters, which can only be used as optional argument for `\includesvg` (`angle` and
`inkscapearea` (param.) `origin`) but not as an option. Now all parameters are set in local context (within a group).
`inkscapedpi` (param.)
`inkscapeopt` (param.)

```

718   \svg@local@param@set{#1}%

```

`svgextension` (param.)

The file suffix used by both packages `svg` and `svg-extract`.

```

width (param.)
height (param.)
distort (param.)
scale (param.)
pretex (param.)
apptex (param.)
draft (param.)
719   \if@svg@ink@latex%
720     \edef\svg@file@suffix[_\svg@file@ext-tex]%
721   \else%
722     \edef\svg@file@suffix[_\svg@file@ext-raw]%
723   \fi%
724   \@onelevel@sanitize\svg@file@suffix%

```

Searching all given paths for the relevant SVG file.

```

725   \svg@get@path{#2}{}%
726   \if@svg@file@found%

```

Running the export with *Inkscape* (if necessary) and checking the required files for graphic inclusion.

```

727     \svg@ink@run%
728     \IfFileExists{\svg@out@base}{}%
729     \@svg@file@foundfalse%
730     \svg@file@missing{\svg@out@base}{\svg@file@base.\svg@file@ext}%
731   }%
732   \if@svg@ink@latex%
733     \IfFileExists{\svg@out@base_tex}{}%
734     \@svg@file@foundfalse%
735     \svg@file@missing{\svg@out@base_tex}{\svg@file@base.\svg@file@ext}%
736   }%
737   \fi%

```

Include the resulting graphic file and maybe extract independent files.

```

738     \if@svg@file@found%
739       \svg@input{\svg@out@base}%
740       \svg@extract{\svg@out@base}%
741     \fi%
742   \else%

```


Raise an error, if the requested SVG file wasn't found.

```
743     \svg@file@missing[\svg@file@ext]{\svg@file@base}{}%  
744     \fi%  
745 \endgroup%  
746 }
```

`\lastpage` (param.) In addition to the automatic finding of the last page, which is included, it can also be given directly as parameter.

```
747 \svg@local@param@def{%  
748   \FamilyCounterKey[.param]{SVG}{lastpage}{svg@param@lastpage}%  
749 }
```

`\angle` (param.) The parameters `angle` and `origin` are defined as pendants to the keys provided by `\includegraphics`.

```
750 \newcommand*\svg@param@angle{0}  
751 \svg@local@param@def{%  
752   \DefineFamilyKey[.param]{SVG}{angle}{%  
753     \ifisdimension{#1\p@}{%  
754       \renewcommand*\svg@param@angle{#1}%  
755       \FamilyKeyStateProcessed%  
756     }{}%  
757   }%  
758 }  
759 \newcommand*\svg@param@origin{c}  
760 \svg@local@param@def{%  
761   \DefineFamilyKey[.param]{SVG}{origin}{c}{%  
762     \renewcommand*\svg@param@origin{#1}%  
763     \FamilyKeyStateProcessed%  
764   }%  
765 }
```

`\includeinkscape` The command `\includeinkscape` can be used for including the export results of *Inkscape*, if this part of the job was done in another way.

```
766 \newcommand*\includeinkscape[2] [] {%  
767   \begingroup%
```

Checking for deprecated commands `\svgwidth` and `\svgscale`.

```
768   \svg@deprecated@param%
```

The given file extension is examined, where a known extension overwrites the current setting for `inkscapeformat`. If there's a suffix `_tex`, the option `inkscapecatex` is set to `true` by default.

```
769   \svg@filename@parse{#2}%  
770   \ifx\filename@ext\relax\else%  
771     \svg@quotes@remove{\filename@ext}%  
772     \expandafter\lowercase\expandafter{%  
773       \expandafter\def\expandafter\filename@ext\expandafter{\filename@ext}%  
774     }%  
775     \def\svg@tempb##1_tex##2\@nil{%  
776       \IfArgIsEmpty{##1}{\def\filename@ext{##1}}%  
777       \ifstr{##2}{_tex}{\@svg@tempwatrue}{\@svg@tempwafalse}%  
778     }%  
779     \@svg@tempwafalse%  
780     \@tfor\svg@tempa:={pdf}{eps}{ps}{png}\do{%  
781       \begingroup%  
782         \expandafter\svg@tempb\filename@ext_tex\@nil%  
783         \svg@extension@parse{\svg@tempa}%  
784         \ifx\filename@ext\relax%  
785           \def\svg@tempb{\endgroup}%
```

```

786         \else%
787         \edef\svg@tempb{%
788         \endgroup%
789         \noexpand\FamilyOptions{SVG}{inkscapeformat=\svg@tempa}%
790         \if@svg@tempswa%
791         \noexpand\FamilyOptions{SVG}{inkscapelatex=true}%
792         \fi%
793         \def\noexpand\filename@base{\filename@base}%
794         \def\noexpand\filename@ext{\filename@ext}%
795         \noexpand\@svg@tempswatrue%
796     }%
797     \fi%
798     \svg@tempb%

```

Break for loop, if valid extension was found.

```

799         \if@svg@tempswa%
800         \@break@tfor%
801         \fi%
802     }%

```

If no valid extension was found, it is set to the specified format and the actual found one is appended to `cssvg.dtx@base`.

```

803         \if@svg@tempswa\else%
804         \svg@extension@parse{\svg@ink@format}%
805         \fi%
806     \fi%

```

`inkscapeformat` (param.) Parameters, which are supported by `\includesvg`, can also be used with `\includeinkscape` even if some of them—more precisely those that control the export with *Inkscape*—don't have an effect at all. Nevertheless, they are set right now in local context (within a group).

```

width (param.)
height (param.)
distort (param.) 807     \svg@local@param@set{#1}%
scale (param.)

```

`pretex` (param.) Searching all given paths for the relevant PDF/EPS file.

```

apptex (param.)
draft (param.) 808     \svg@get@path[\svg@ink@format]{\filename@area\filename@base}{\svg@out@path}%
lastpage (param.) 809     \if@svg@file@found%
angle (param.)
origin (param.)

```

Checking the required files for graphic inclusion.

```

810     \edef\svg@out@name{\svg@file@name}%
811     \edef\svg@out@base{\svg@file@path\svg@file@name.\svg@ink@format}%
812     \if@svg@ink@latex%
813         \IfFileExists{\svg@out@base_tex}{}{%
814             \@svg@file@foundfalse%
815             \svg@file@missing{\svg@out@base_tex}{\svg@out@base}%
816         }%
817     \fi%

```

Include the resulting graphic file and maybe extract independent files.

```

818         \if@svg@file@found%
819         \svg@input{\svg@out@base}%
820         \svg@extract{\svg@out@base}%
821         \fi%
822     \else%

```

Raise an error, if the requested PDF/EPS file wasn't found.

```

823         \svg@file@missing[\svg@ink@format]{\svg@file@base}{\svg@out@path}%
824         \fi%
825     \endgroup%
826 }

```

B.5. Auxiliary macros

`\svg@deprecated@param` This macro checks, if `\svgwidth` or `\svgscale` are defined. In this case, the given values are passed to the correlating parameters and a warning is raised.

```

827 \newcommand*\svg@deprecated@param{%
828   \@svg@tempswafalse%
829   \ifx\svgwidth\@undefined\else%
830     \edef\svg@tempa{\noexpand\FamilyOptions{SVG}{width=\svgwidth}}%
831     \svg@tempa%
832     \@svg@tempswatrue%
833   \fi%
834   \ifx\svgscale\@undefined\else%
835     \edef\svg@tempa{\noexpand\FamilyOptions{SVG}{scale=\svgscale}}%
836     \svg@tempa%
837     \@svg@tempswatrue%
838   \fi%
839   \if@svg@tempswa%
840     \PackageWarning{svg}{%
841       You should specify the image size with parameters\MessageBreak%
842       ‘width’ and ‘height’ or ‘scale’ instead of using\MessageBreak%
843       ‘\string\svgscale’ or ‘\string\svgwidth’%
844     }%
845     \let\svgwidth\@undefined%
846     \let\svgscale\@undefined%
847   \fi%
848 }
```

`\svg@ink@run` The command, which performs the call of *Inkscape* via `\ShellEscape`.
`\if@svg@ink@run`

```

849 \newif\if@svg@ink@run
850 \newcommand*\svg@ink@run{%
851   \ifnum\svg@ink@mode>\z@\relax%
852     \begingroup%
```

If the mode for *inkscape* was set to *forced*, *Inkscape* will be called in any case. Otherwise, some checks are performed to detect, if a run of *Inkscape* is actually necessary.

```

853     \@svg@ink@runtrue%
854     \ifnum\svg@ink@mode=\tw@\relax\else%
```

This is the case when the SVG file is newer than the corresponding exported file, or if the latter isn't present at all.

```

855     \svg@iffilenewer{\svg@file@base.\svg@file@ext}{\svg@out@base}{}%
856     \@svg@ink@runfalse%
857   }%
```

The same is true, when the associated L^AT_EX file is missing. But when this file already exists, maybe the user did some changes to this file. In this case, overwriting this file is maybe not intended.

```

858     \if@svg@ink@latex%
859     \IfFileExists{\svg@out@base_tex}{%
860       \ifnum\pdf@shellescape=\@one\relax\if@svg@ink@run%
861         \svg@iffilenewer{\svg@out@base_tex}{\svg@out@base}{%
862           \@svg@ink@runfalse%
863           \svg@quotes@remove[\svg@out@base]{\svg@tempa}%
864           \PackageWarning{svg}{%
865             Since the encountered filedate of file\MessageBreak%
866             ‘\svg@tempa_tex’ is newer than \MessageBreak%
867             ‘\svg@tempa’ it’s supposed that\MessageBreak%
868             you customized this file. To avoid an accidental\MessageBreak%
869             overwriting of this file, the Inkscape export\MessageBreak%
870             won’t be done. If you want to overwrite the\MessageBreak%
871             existing file please choose the parameter\MessageBreak%
```

```

872             'inkscape=force'%
873             }%
874         }{}%
875         \fi\fi%
876     }{\@svg@ink@runtrue}%
877     \fi%
878     \fi%

```

If all checks were positive, the export with *Inkscape* can be done in case `--shell-escape` is enabled.

```

879     \if@svg@ink@run%
880     \ifnum\pdf@shellescape=\@one\relax%

```

For exporting PNG files, the used density is set to 300dpi, if no value was given.

```

881     \ifx\svg@ink@dpi\relax%
882     \ifstr{\svg@ink@format}{png}{%
883     \FamilyOptions{SVG}{inkscapedpi=300}%
884     }{}%
885     \fi%
886     \PackageInfo{svg}{%
887     Calling Inkscape%
888     \ifx\svg@ink@opt\@empty\else%
889     \space with added options '\svg@ink@opt'%
890     \fi%
891     }%

```

Executing *Inkscape* on command line. Afterwards, the export results are moved into the given output path.

```

892     \svg@quotes@remove[\svg@file@base]{\svg@tempa}%
893     \svg@quotes@remove[\svg@out@name]{\svg@tempb}%
894     \ShellEscape{\svg@ink@cmd{\svg@tempa}{\svg@tempb}}%
895     \IfFileExists{\svg@out@name.\svg@ink@format}{%
896     \edef\svg@tempb{\svg@tempb.\svg@ink@format}%
897     \svg@quotes@remove{\svg@out@base}%
898     \svg@shell@mkdir{\svg@out@path}%
899     \svg@shell@move{\svg@tempb}{\svg@out@base}%
900     \if@svg@ink@latex%
901     \svg@shell@move{\svg@tempb_tex}{\svg@out@base_tex}%
902     \fi%
903     }{}%
904     \PackageWarning{svg}{%
905     The export with Inkscape failed for file\MessageBreak%
906     '\svg@tempa.\svg@file@ext'\MessageBreak%
907     Troubleshooting: Please check in the log file how\MessageBreak%
908     the invocation of Inkscape took place and try to\MessageBreak%
909     execute it yourself in the terminal%
910     }%
911     }%

```

If `--shell-escape` wasn't enabled, a warning is issued.

```

912     \else%
913     \svg@quotes@remove[\svg@file@base]{\svg@tempa}%
914     \PackageWarning{svg}{%
915     You didn't enable 'shell escape' (or 'write18')\MessageBreak%
916     so it wasn't possible to launch the Inkscape export\MessageBreak%
917     for '\svg@tempa.\svg@file@ext'%
918     }%
919     \fi%
920     \fi%
921     \endgroup%
922     \fi%
923 }

```

`\svg@ink@cmd` The actual call of *Inkscape* at command line.

```
924 \newcommand*\svg@ink@cmd[2]{%
925   \svg@ink@exe\space-z\space\svg@ink@area\space%
926   \ifx\svg@ink@dpi\relax\else--export-dpi=\svg@ink@dpi\space\fi%
927   \if\svg@ink@latex--export-latex\space\fi%
928   \svg@ink@opt\space%
929   --file="#1.\svg@file@ext"\space%
930   --export-\svg@ink@format="#2.\svg@ink@format"\space%
931 }
```

`\svg@get@lastpage` This macro is used to circumvent the multiple pages bug for PDF files of *Inkscape* 0.91, when the the L^AT_EX export was enabled. For this purpose, the total page number is read from the PDF file.

```
932 \newcommand*\svg@get@lastpage[1]{%
933   \ifstr{\svg@ink@format}{pdf}{-%
934     \begingroup%
935     \@tempcnta=\m@ne\relax%
936     \ifx\XeTeXpdfpagecount\@undefined%
937       \ifpdf%
938         \ifx\pdfximage\@undefined%
939           \ifx\saveimageresource\@undefined\else%
940             \saveimageresource{#1}%
941             \@tempcnta=\lastsavedimageresourcepages\relax%
942             \fi%
943           \else%
944             \pdfximage{#1}%
945             \@tempcnta=\pdflastximagepages\relax%
946             \fi%
947           \fi%
948         \else%
949           \@tempcnta=\XeTeXpdfpagecount#1\relax%
950           \fi%
951         \ifnum\@tempcnta=\m@ne\relax%
952           \PackageWarning{svg}{%
953             It wasn't possible to detect the last page\MessageBreak%
954             of '#1'%
955           }%
956         \else%
957           \PackageInfo{svg}{Last page of '#1' is \the\@tempcnta}%
958           \fi%
959         \edef\svg@tempa{%
960           \endgroup%
961           \noexpand\FamilyOptions{SVG}{lastpage=\the\@tempcnta}%
962         }%
963       \svg@tempa%
964     }{-%
965 }
```

`\svg@wrn@scale` The option `scale` respectively the parameter `scale` is only considered if the size was not specified.

```
966 \newcommand*\svg@wrn@scale{%
967   \ifdim\dimexpr\svg@param@scale\p@\relax=\p@\relax\else%
968     \@svg@tempswafalse%
969     \ifdim\svg@param@width>\z@\relax%
970       \@svg@tempswatruetrue%
971       \fi%
972     \ifdim\svg@param@height>\z@\relax%
973       \@svg@tempswatruetrue%
974       \fi%
975     \if@svg@tempswa%
976       \PackageWarning{svg}{%
977         The parameter 'scale' is only considered if neither\MessageBreak%
```

```

978         'width' nor 'height' are specified%
979     }%
980     \fi%
981 \fi%
982 }

```

`\svg@input` With `\svg@@input` the export results of *Inkscape* are included. The macro `\svg@input` is defined in order to realize the option `exclude` for package **svg-extract**. The macro `\svg@set@input@path` is called to support commands like `\input{(tex filename)}` within SVG files.

```

983 \newcommand*\svg@input{\svg@@input}
984 \newcommand*\svg@@input[2][{}]{%
985     \IfArgIsEmpty{#1}{\svg@local@param@set{#1}}%
986     \svg@set@input@path%
987     \if@svg@draft%
988         \@svg@ink@latexfalse%
989     \fi%

```

In order to support file names with multiple dots, the second argument is parsed and only the part after the last dot is stroed in `\svg@tempb` as extension. Everything before is stored in `\svg@tempa`.

```

990 \def\svg@tempb##1.##2\@nil{%
991     \IfArgIsEmpty{##2}{%
992         \def\svg@tempb{##1}%
993     }{%
994         \edef\svg@tempa{\svg@tempa.##1}%
995         \svg@tempb##2\@nil%
996     }%
997 }%
998 \edef\svg@tempa{%
999     \def\noexpand\svg@tempa{}%
1000     \noexpand\svg@tempb#2.\noexpand\@nil%
1001 }%
1002 \svg@tempa%

```

Afterwards `\svg@tempa` is defined with the file name within enclosing braces followed by the extension—only if the file name itself contains any dots—and `\svg@tempb` holds the original file name plus extension without enclosing braces.

```

1003 \svg@remove@leadingchar.\svg@tempa%
1004 \begingroup%
1005     \expandafter\filename@parse\expandafter{\svg@tempa}%
1006     \edef\svg@tempa{%
1007         \endgroup%
1008         \ifx\filename@ext\relax%
1009             \edef\noexpand\svg@tempa{\svg@tempa.\svg@tempb}%
1010         \else%
1011             \edef\noexpand\svg@tempa{{\svg@tempa}.\svg@tempb}%
1012         \fi%
1013     }%
1014 \svg@tempa%
1015 \edef\svg@tempb{#2}%

```

If the export with *Inkscape* was done with L^AT_EX support enabled, the corresponding file will be used together with `\input`. The necessary patches to environment `picture` as well as command `\includegraphics` are made beforehand with `\svg@patches`.

```

1016 \if@svg@ink@latex%
1017     \svg@patches{\svg@tempa}%
1018     \ifnum\value{svg@param@lastpage}=\z@\relax%
1019         \expandafter\svg@get@lastpage\expandafter{\svg@tempb}%
1020     \fi%
1021 \edef\svg@tempa{%

```

```

1022     \ifx\svg@param@pretex\relax\else%
1023         \noexpand\svg@param@pretex%
1024     \fi%
1025     \noexpand\input{\svg@tempb_tex}%
1026     \ifx\svg@param@apptex\relax\else%
1027         \noexpand\svg@param@apptex%
1028     \fi%
1029 }%

```

If `distort=true` is desired, the input is resized with `\resizebox*`.

```

1030     \ifsvg@param@distort%
1031         \def\svg@tempb{\resizebox*{\svg@param@width}{\svg@param@height}}%
1032     \else%
1033         \let\svg@tempb\@firstofone%
1034     \fi%
1035     \sbox\svg@box{\svg@tempb{\svg@tempa}}%

```

If a rotation angle was given, the input is done within `\rotatebox`.

```

1036     \ifdim\dimexpr\svg@param@angle\p@\relax=\z@\relax%
1037         \let\svg@tempb\@firstofone%
1038     \else%
1039         \edef\svg@tempb{%
1040             \noexpand\rotatebox[origin=\svg@param@origin]{\svg@param@angle}%
1041         }%
1042     \fi%
1043     \svg@tempb{\usebox\svg@box}%
1044 \else%

```

If the export with *Inkscape* was done without \LaTeX support, the resulting graphic file will be included with `\includegraphics`.

```

1045     \svg@wrn@scale%
1046     \edef\svg@tempb{%
1047         draft\ifsvg@draft\else=false\fi,%
1048         scale=\svg@param@scale,%
1049         keepaspectratio\ifsvg@param@distort=false\fi%
1050     }%
1051     \ifdim\svg@param@height>\z@\relax%
1052         \edef\svg@tempb{\svg@tempb,height=\svg@param@height}%
1053     \fi%
1054     \ifdim\svg@param@width>\z@\relax%
1055         \edef\svg@tempb{\svg@tempb,width=\svg@param@width}%
1056     \fi%
1057     \ifdim\dimexpr\svg@param@angle\p@\relax=\z@\relax\else%
1058         \edef\svg@tempb{%
1059             \svg@tempb,origin=\svg@param@origin,angle=\svg@param@angle%
1060         }%
1061     \fi%
1062     \expandafter\includegraphics\expandafter[\svg@tempb]{\svg@tempa}%
1063 \fi%
1064 }

```

B.6. Patches

```

\svg@patches
\svg@picture@saved
\svg@includegraphics@saved

```

For including the export results from *Inkscape* with \LaTeX support enabled, there are some patches necessary for environment `picture` and `\includegraphics`. Those patches are done with `\svg@patches`.

```

1065 \newcommand*\svg@picture@saved{}
1066 \let\svg@picture@saved\picture
1067 \newcommand*\svg@includegraphics@saved{}
1068 \let\svg@includegraphics@saved\includegraphics
1069 \newcommand*\svg@patches[1]{%

```

```

1070 \let\picture\svg@picture@patched%
1071 \let\includegraphics\svg@includegraphics@patched%
1072 \edef\svg@includegraphics@file{#1}%
1073 }

```

`\svg@picture@patched` In order to provide the possibility specify the desired width of a graphic, the appropriate `\unitlength` is calculated at the beginning of the picture environment.

```

1074 \newcommand*\svg@picture@patched{}
1075 \newcommand*\svg@picture@patched{}
1076 \long\def\svg@picture@patched#1{\svg@picture@patched@#1}
1077 \def\svg@picture@patched@(#1,#2){%
1078 \svg@width@scale%

```

If a desired height is present, the resulting `\unitlength` is calculated with the ratio of the coordinates of the `picture` environment given as arguments for x- and y-direction by using `\Gscale@div`. With this factor, `\unitlength`—which is connected to the x-coordinate—can be scaled in a suitable manner.

```

1079 \ifdim\svg@param@height>\z@\relax%
1080 \Gscale@div\svg@tempa{#1\p@}{#2\p@}%
1081 \setlength\unitlength{\svg@param@height}%
1082 \setlength\unitlength{\svg@tempa\unitlength}%
1083 \ifdim\svg@param@width>\z@\relax%
1084 \ifdim\unitlength>\svg@param@width\relax%
1085 \setlength\unitlength{\svg@param@width}%
1086 \fi%
1087 \fi%
1088 \else%

```

If no height is given, `\unitlength` can be set easily.

```

1089 \ifdim\svg@param@width>\z@\relax%
1090 \setlength\unitlength{\svg@param@width}%
1091 \else%
1092 \setlength\unitlength{\svg@param@scale\unitlength}%
1093 \fi%
1094 \fi%

```

After setting `\unitlength`, the `picture` environment can be called with its original definition.

```

1095 \svg@picture@saved(#1,#2)%
1096 }

```

`\svg@includegraphics@patched` `\svg@includegraphics@file` The patch to `\includegraphics` is meant to dissolve the *Inkscape* bug concerning the inclusion of more PDF pages than actually are existing.

The given optional parameters to `\includegraphics` are processed and the counter `svg@param@currpage` is set to the value of a given page. The value of parameter `width` is ignored.

```

1097 \DefineFamily{SVGpatch}
1098 \DefineFamilyMember{SVGpatch}
1099 \newcounter{svg@param@currpage}
1100 \setcounter{svg@param@currpage}{\m@ne}
1101 \FamilyCounterKey{SVGpatch}{page}{\svg@param@currpage}
1102 \DefineFamilyKey{SVGpatch}{width}{\FamilyKeyStateProcessed}
1103 \newcommand*\svg@includegraphics@file{}
1104 \newcommand*\svg@includegraphics@patched[2][ ]{%
1105 \FamilyOptions{SVGpatch}{#1}%

```

If option `lastpage` was set to `false`, each page is included—even if it doesn't exist, which may cause errors.

```

1106 \ifnum\value{svg@param@lastpage}<\z@\relax%
1107 \FamilySetCounter{SVGpatch}{page}{svg@param@currpage}{%

```



```

1108     \the\value{svg@param@lastpage}%
1109     }%
1110 \fi%

```

Only if counter `svg@param@lastpage` is smaller than `svg@param@currpage`, pages are included, where `svg@param@lastpage` was either given as a number with parameter `lastpage` or was automatically calculated with `\svg@get@lastpage`.

```

1111 \ifnum\value{svg@param@currpage}>\value{svg@param@lastpage}\relax\else%

```

A page is included with the original definition of `\includegraphics`. All optional parameters are passed.

```

1112 \svg@includegraphics@saved[#{1}]{\svg@includegraphics@file}%
1113 \fi%
1114 }

```

C. Extracting independent graphic files with `svg-extract`

C.1. Options

For package `svg-extract` the user interface is extended. The following options can either be set with `\svgsetup` or be used as local optional parameters for `\includesvg` and `\includeinkscape`.

`\svg@dummy@key` If package `svg-extract` wasn't loaded, the following options are defined for package `svg` in order to raise a warning message. Primarily this is done for compatibility reasons.

```

1115 (*base)
1116 \DefineFamilyMember[.dummy]{SVG}
1117 \newcommand*\svg@dummy@key[2][ ]{%
1118   \@ifpackageloaded{svg-extract}{}{%
1119     \IfArgIsEmpty{#1}{%
1120       \DefineFamilyKey[.dummy]{SVG}{#2}{%
1121         \PackageWarning{svg}{%
1122           The option key '#2' can only\MessageBreak%
1123           be used with package 'svg-extract', but\MessageBreak%
1124           you didn't load it%
1125         }%
1126         \FamilyKeyStateProcessed%
1127       }%
1128     }%
1129     \DefineFamilyKey[.dummy]{SVG}{#2}[{#1}]{%
1130       \PackageWarning{svg}{%
1131         The option key '#2' can only\MessageBreak%
1132         be used with package 'svg-extract', but\MessageBreak%
1133         you didn't load it%
1134       }%
1135       \FamilyKeyStateProcessed%
1136     }%
1137   }%

```

Before package `svg-extract` the given key `#2` of family member `.dummy` is relaxed.

```

1138   \AfterPackage{svg-extract}{\RelaxFamilyKey[.dummy]{SVG}{#2}}%
1139 }%
1140 }
1141 </base>

```

C.1.1. Controlling the extract process

`extract` (opt.) With option `extract` it can be controlled, if the extraction of independent graphic files
`\if@svgx@run` should be done.

```
1142 < *base >
1143 \svg@dummys@key[true]{extract}
1144 < /base >
1145 < *extract >
1146 \newif\if@svgx@run
1147 \DefineFamilyKey{SVG}{extract}[true]{%
1148   \lowercase{\def\svg@tempa{#1}}%
1149   \FamilySetNumerical{SVG}{extract}{svg@tempa}{%
1150     {false}{0},{off}{0},{no}{0},%
1151     {true}{1},{on}{1},{yes}{1},{onlynewer}{1},{newer}{1},%
1152     {overwrite}{1},{force}{1},{forced}{1},%
1153     {pdf}{2},{eps}{3},{ps}{4}}%
1154 }{svg@tempa}%
1155 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1156   \ifcase\svg@tempa\relax% false
1157     \svgx@runfalse%
1158   \or% true
1159     \svgx@runtrue%
1160   \or% pdf
1161     \FamilyOptions{SVG}{extractformat=pdf}%
1162   \or% eps
1163     \FamilyOptions{SVG}{extractformat=eps}%
1164   \or% ps
1165     \FamilyOptions{SVG}{extractformat=ps}%
1166   \fi%
1167 \fi%
1168 }
1169 < /extract >
```

`on` (opt.) Package options which can be used to switch functionality on or off during the loading of
`off` (opt.) package **svg-extract**.

```
1170 < *extract >
1171 \DeclareOption{on}{\FamilyOptions{SVG}{extract=true}}
1172 \DeclareOption{off}{\FamilyOptions{SVG}{extract=false}}
1173 < /extract >
```

`extractformat` (opt.) Option `extractformat` controls the output format (pdf/eps/ps). It is set to pdf or, if dvi
`\svgx@format` output could be detected, to eps during initialization.

```
pdf (opt.)
eps (opt.)
1174 < *base >
1175 \svg@dummys@key{extractformat}
1176 \svg@dummys@key[true]{pdf}
1177 \svg@dummys@key[true]{eps}
1178 < /base >
1179 < *extract >
1180 \newcommand*\svgx@format{pdf}
1181 \ifxetex\else\ifpdf\else
1182   \renewcommand*\svgx@format{eps}
1183 \fi\fi
1184 \DefineFamilyKey{SVG}{extractformat}{%
1185   \lowercase{\edef\svgx@format{#1}}%
1186   \FamilyKeyStateProcessed%
1187 }
1188 \DefineFamilyKey{SVG}{pdf}[true]{%
1189   \FamilySetBool{SVG}{pdf}{@svg@tempa}{#1}%
1190   \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1191     \if@svg@tempa%
1192       \svgx@ifinlist{pdf}{\svgx@format}{}%
1193       \edef\svgx@format{\svgx@format,pdf}%

```

```

1194     }%
1195     \svg@deprecated@key{pdf}{extractformat={\svgx@format}}%
1196     \else%
1197         \FamilyKeyStateUnknownValue%
1198     \fi%
1199 \fi%
1200 }
1201 \DefineFamilyKey{SVG}{eps}[true]{%
1202 \FamilySetBool{SVG}{eps}{@svg@tempswa}{#1}%
1203 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1204 \if@svg@tempswa%
1205 \svgx@ifinlist{eps}{\svgx@format}{}%
1206 \edef\svgx@format{\svgx@format,eps}%
1207 }%
1208 \svg@deprecated@key{eps}{extractformat={\svgx@format}}%
1209 \else%
1210 \FamilyKeyStateUnknownValue%
1211 \fi%
1212 \fi%
1213 }
1214 </extract>

```

`extractpreamble` (opt.) For the extraction process, a preamble is necessary for a separate auxiliary L^AT_EX file.
`preamble` (opt.) By default, the preamble of the main document is used, which end is detected at
`\svgx@preamble` `\begin{document}`.

```

extractpreambleend (opt.)
end (opt.)
\svgx@endpreamble
1215 (*base)
1216 \svg@dummy@key{extractpreamble}
1217 \svg@dummy@key{preamble}
1218 \svg@dummy@key{extractpreambleend}
1219 \svg@dummy@key{end}
1220 </base>
1221 (*extract)
1222 \newcommand*\svgx@preamble{\jobname.\svgx@latex@ext}%
1223 \DefineFamilyKey{SVG}{extractpreamble}{%
1224 \renewcommand*\svgx@preamble{#1}%
1225 \FamilyKeyStateProcessed%
1226 }
1227 \DefineFamilyKey{SVG}{preamble}{%
1228 \svg@deprecated@key[svg-extract]{preamble=#1}{extractpreamble=#1}%
1229 }
1230 \newcommand*\svgx@endpreamble{}
1231 \expandafter\def\expandafter\svgx@endpreamble\expandafter{%
1232 \cscname begin\endcscname{document}%
1233 }
1234 \DefineFamilyKey{SVG}{extractpreambleend}{%
1235 \renewcommand*\svgx@endpreamble{#1}%
1236 \FamilyKeyStateProcessed%
1237 }
1238 \DefineFamilyKey{SVG}{end}{%
1239 \svg@deprecated@key[svg-extract]{end=#1}{extractpreambleend=#1}%
1240 }
1241 </extract>

```

`extractruns` (opt.) With this option, the number of L^AT_EX runs for the separate auxiliary file can be set.
`svgx@runs` (counter)

```

1242 (*base)
1243 \svg@dummy@key{extractruns}
1244 </base>
1245 (*extract)
1246 \newcounter{svgx@runs}
1247 \DefineFamilyKey{SVG}{extractruns}{%
1248 \FamilySetCounter{SVG}{extractruns}{svgx@runs}{#1}%
1249 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1250 \ifnum\value{svgx@runs}<\@ne\relax%

```

```

1251     \PackageWarning{svg-extract}{%
1252       The count for runs has to be at least one%
1253     }%
1254     \FamilySetCounter{SVG}{extractruns}{svgx@runs}{\@ne}%
1255     \fi%
1256   \fi%
1257 }
1258 </extract>

latexexe (opt.) The command and facultative options for the LATEX call of the separate auxiliary file. The
pdflatex (opt.) default is set according to the currently used compiler.
\svgx@latex@exe
  latexext (opt.) 1259 <(*base)
\svgx@latex@ext
  latexopt (opt.) 1260 \svg@dumy@key{latexexe}
1261 \svg@dumy@key{pdflatex}
\svgx@latex@opt 1262 \svg@dumy@key{latexext}
1263 \svg@dumy@key{latexopt}
1264 </base>
1265 <(*extract)
1266 \ifxetex
1267   \newcommand*\svgx@latex@exe{xelatex}
1268 \else\ifluatex
1269   \ifpdf
1270     \newcommand*\svgx@latex@exe{lualatex}
1271   \else
1272     \newcommand*\svgx@latex@exe{lualatex --output-format=dvi}
1273   \fi
1274 \else\ifpdf
1275   \newcommand*\svgx@latex@exe{pdflatex}
1276 \else
1277   \newcommand*\svgx@latex@exe{latex}
1278 \fi\fi\fi
1279 \DefineFamilyKey{SVG}{latexexe}{%
1280   \renewcommand*\svgx@latex@exe{#1}%
1281   \FamilyKeyStateProcessed%
1282 }
1283 \DefineFamilyKey{SVG}{pdflatex}{%
1284   \svg@deprecated@key[svg-extract]{pdflatex=#1}{latexexe=#1}%
1285 }
1286 \newcommand*\svgx@latex@ext{tex}
1287 \DefineFamilyKey{SVG}{latexext}{%
1288   \renewcommand*\svgx@latex@ext{#1}%
1289   \FamilyKeyStateProcessed%
1290 }
1291 \newcommand*\svgx@latex@opt{}
1292 \DefineFamilyKey{SVG}{latexopt}{%
1293   \renewcommand*\svgx@latex@opt{#1}%
1294   \FamilyKeyStateProcessed%
1295 }
1296 </extract>

dvipsopt (opt.) Options and macros for calling convert commands, which are supplied by most LATEX 2ε distri-
\svgx@dvips@exe butions. These are used to generate all files, which are supported by option extractformat,
\svgx@dvips@opt as they don't need an additional application.
pstoepsopt (opt.)
\svgx@pstoeps@exe 1297 <(*base)
\svgx@pstoeps@opt 1298 \svg@dumy@key{dvipsopt}
1299 \svg@dumy@key{pstoepsopt}
pstopdfopt (opt.) 1300 \svg@dumy@key{pstopdfopt}
\svgx@pstopdf@exe 1301 \svg@dumy@key{pdfstoepsopt}
\svgx@pstopdf@opt 1302 \svg@dumy@key{pdfstopdfopt}
1303 \svg@dumy@key{pdftops}
pdftoepsopt (opt.) 1304 </base>
\svgx@pdftoeps@exe 1305 <(*extract)
\svgx@pdftoeps@opt 1306 \newcommand*\svgx@dvips@exe{dvips}
pdftops (opt.)

```

```

1307 \newcommand*{svgx@dvdvips@opt{}}
1308 \DefineFamilyKey{SVG}{dvdvipsopt}{%
1309   \renewcommand*{svgx@dvdvips@opt{#1}}%
1310   \FamilyKeyStateProcessed%
1311 }
1312 \newcommand*{svgx@pstoeps@exe{ps2eps}}
1313 \newcommand*{svgx@pstoeps@opt{-B -C}}
1314 \DefineFamilyKey{SVG}{pstoepsopt}{%
1315   \renewcommand*{svgx@pstoeps@opt{#1}}%
1316   \FamilyKeyStateProcessed%
1317 }
1318 \newcommand*{svgx@pstopdf@exe{ps2pdf}}
1319 \newcommand*{svgx@pstopdf@opt{}}
1320 \DefineFamilyKey{SVG}{pstopdfopt}{%
1321   \renewcommand*{svgx@pstopdf@opt{#1}}%
1322   \FamilyKeyStateProcessed%
1323 }
1324 \newcommand*{svgx@pdftoeps@exe{pdftops -eps}}
1325 \newcommand*{svgx@pdftoeps@opt{}}
1326 \DefineFamilyKey{SVG}{pdftoepsopt}{%
1327   \renewcommand*{svgx@pdftoeps@opt{#1}}%
1328   \FamilyKeyStateProcessed%
1329 }
1330 \newcommand*{svgx@pdftops@exe{pdftops}}
1331 \newcommand*{svgx@pdftops@opt{}}
1332 \DefineFamilyKey{SVG}{pdftopsopt}{%
1333   \renewcommand*{svgx@pdftops@opt{#1}}%
1334   \FamilyKeyStateProcessed%
1335 }
1336 \DefineFamilyKey{SVG}{pdftops}{%
1337   \PackageWarning{#1}{%
1338     The option key 'pdftops' is deprecated.\MessageBreak%
1339     You should use either 'pdftoepsopt' or\MessageBreak%
1340     'pdftopsopt' instead. See the manual for\MessageBreak%
1341     more. Nothing was done%
1342   }}%
1343   \FamilyKeyStateProcessed%
1344 }
1345 </extract>

```

C.1.2. Invoking external application for graphic conversion

Besides the use of a conversion tool supplied by L^AT_EX 2_ε, the applications *ImageMagick* and *Ghostscript* can be used for converting graphics.

```

convert (opt.) The option convert can be used to define, which of both applications should be use.
\if@svgx@cnv@run ImageMagick is set by default.
\svgx@cnv@cmd
1346 (*base)
1347 \svg@dummy@key[true]{convert}
1348 </base>
1349 (*extract)
1350 \newif\if@svgx@cnv@run
1351 \newcommand*{svgx@cnv@cmd}{
1352 \DefineFamilyKey{SVG}{convert}[true]{%
1353   \FamilySetNumerical{SVG}{convert}{svg@tempa}{%
1354     {false}{0},{off}{0},{no}{0},%
1355     {true}{1},{on}{1},{yes}{1},{onlynewer}{1},{newer}{1},%
1356     {overwrite}{1},{force}{1},{forced}{1},%
1357     {magick}{2},{imagemagick}{2},{convert}{2},%
1358     {gs}{3},{ghostscript}{3},%
1359     {gs64}{4},{ghostscript64}{4},%
1360     {gs32}{5},{ghostscript32}{5}%
1361   }{#1}%

```

```

1362 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1363 \ifcase\svg@tempa\relax% false
1364 \@svgx@cnv@runfalse%
1365 \or% true
1366 \@svgx@cnv@runtrue%
1367 \or% magick
1368 \@svgx@cnv@runtrue%
1369 \renewcommand*\svgx@cnv@cmd{\svgx@magick@cmd}%
1370 \or% gs
1371 \@svgx@cnv@runtrue%
1372 \renewcommand*\svgx@cnv@cmd{\svgx@gs@cmd}%
1373 \or% gs64
1374 \@svgx@cnv@runtrue%
1375 \renewcommand*\svgx@cnv@cmd{\svgx@gs@cmd}%
1376 \svgx@onlywindows{%
1377 \renewcommand*\svgx@gs@exe{gswin64c}%
1378 }%
1379 \or% gs32
1380 \@svgx@cnv@runtrue%
1381 \renewcommand*\svgx@cnv@cmd{\svgx@gs@cmd}%
1382 \svgx@onlywindows{%
1383 \renewcommand*\svgx@gs@exe{gswin32c}%
1384 }%
1385 \fi%

```

In version v1.0 the option `convert` was used to set both the executable and options for the conversion application, meant for the usage of *ImageMagick*. This is taken into account here.

```
1386 \else%
```

Same doing like with option `inkscape`.

```

1387 \def\svg@tempa##1-##2\@nil{%
1388 \IfArgsEmpty{##2}{\def\svg@tempb{}}{%
1389 \def\svg@tempa##1###1\@nil{\def\svg@tempb{###1}}%
1390 \svg@tempa#1\@nil%
1391 }%
1392 \def\svg@tempa{##1}%
1393 }%
1394 \svg@tempa#1-\@nil%
1395 \PackageWarning{svg-extract}{%
1396 Setting the executable%
1397 \ifx\svg@tempb\@empty\else%
1398 \space and associated options%
1399 \fi%
1400 \MessageBreak%
1401 for ImageMagick should be done with options\MessageBreak%
1402 'magickexe=\svg@tempa'%
1403 \ifx\svg@tempb\@empty\else%
1404 \MessageBreak and 'magicksetting' and/or 'magickoperator'%
1405 \fi.\MessageBreak%
1406 Nevertheless, this was done by now%
1407 \ifx\svg@tempb\@empty\else%
1408 , whereby \MessageBreak 'magicksetting=\svg@tempb' was used%
1409 \fi%
1410 }%
1411 \FamilyOptions{SVG}{convert=magick}%
1412 \edef\svg@tempa{%
1413 \noexpand\FamilyOptions{SVG}{magickexe=\svg@tempa}%
1414 \ifx\svg@tempb\@empty\else%
1415 \noexpand\FamilyOptions{SVG}{magicksetting=\svg@tempb}%
1416 \fi%
1417 }%
1418 \svg@tempa%
1419 \fi%

```

```

1420 }
1421 </extract>

convertformat (opt.) Option convertformat controls the output format for converted files. It is set to png by
\svgx@cnv@format default.
png (opt.)
1422 (*base)
1423 \svg@dummy@key{convertformat}
1424 \svg@dummy@key[true]{png}
1425 </base>
1426 (*extract)
1427 \newcommand*\svgx@cnv@format{png}
1428 \DefineFamilyKey{SVG}{convertformat}{%
1429 \lowercase{\edef\svgx@cnv@format{#1}}}%
1430 \ifx\svgx@cnv@format\@empty\else%
1431 \@svgx@cnv@runtrue%
1432 \fi%
1433 \FamilyKeyStateProcessed%
1434 }
1435 \DefineFamilyKey{SVG}{png}[true]{%
1436 \FamilySetBool{SVG}{png}{@svg@tempswa}{#1}%
1437 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1438 \ifsvg@tempswa%
1439 \svgx@ifinlist{png}{\svgx@cnv@format}{-}{%
1440 \edef\svgx@cnv@format{\svgx@cnv@format,png}%
1441 }%
1442 \svg@deprecated@key{png}{convertformat={\svgx@cnv@format}}}%
1443 \else%
1444 \FamilyKeyStateUnknownValue%
1445 \fi%
1446 \fi%
1447 }
1448 </extract>

convertdpi (opt.) The option convertdpi is meant to define the used density during the conversion process. It
convertdensity (opt.) can be set either for all designated output formats or targeted for a specific format. It's also
\svgx@cnv@dpi possible to use something like 500x300. Given values are resolved by \svgx@cnv@get@dpi.
It's used like convertdpi=300 or convertdpi={png=600} If the option is used for a specific
or for all output formats is recognized by \svgx@ifkeyandval.

1449 (*base)
1450 \svg@dummy@key{convertdpi}
1451 \svg@dummy@key{convertdensity}
1452 </base>
1453 (*extract)
1454 \newcommand*\svgx@cnv@dpi{
1455 \let\svgx@cnv@dpi\relax
1456 \DefineFamilyKey{SVG}{convertdpi}{%
1457 \FamilyKeyStateUnknownValue%
1458 \svgx@ifkeyandval{#1}{%
1459 \svgx@cnv@get@dpi{##2}%
1460 \ifx\svg@tempa\relax\else%
1461 \expandafter\edef\csname svgx@cnv@dpi@##1\endcsname{\svg@tempa}%
1462 \FamilyKeyStateProcessed%
1463 \fi%
1464 }{%
1465 \svgx@cnv@get@dpi{##1}%
1466 \ifx\svg@tempa\relax\else%
1467 \edef\svgx@cnv@dpi{\svg@tempa}%
1468 \FamilyKeyStateProcessed%
1469 \fi%
1470 }%
1471 }
1472 \DefineFamilyKey{SVG}{convertdensity}{\FamilyOptions{SVG}{convertdpi=#1}}
1473 </extract>

```

magickexe (opt.) Setting the command including maybe the path to **ImageMagick**. The keys `magicksetting` and `magickoperator` should be used to add optional arguments before (*Settings*) or after (*Operators*) the input file. They can either be set for all or a specific output format as like option `convertdpi`. For this `\svgx@setformatkey` is used.

```

\svgx@magick@exe
magicksetting (opt.)
\svgx@magick@set
magickoperator (opt.)
\svgx@magick@opr
1474 (*base)
1475 \svg@dummy@key{magickexe}
1476 \svg@dummy@key{magicksetting}
1477 \svg@dummy@key{magickoperator}
1478 (/base)
1479 (*extract)
1480 \newcommand*\svgx@magick@exe{}
1481 \DefineFamilyKey{SVG}{magickexe}{%
1482   \renewcommand*\svgx@magick@exe{#1}%
1483   \FamilyKeyStateProcessed%
1484 }
1485 \newcommand*\svgx@magick@set{}
1486 \DefineFamilyKey{SVG}{magicksetting}{%
1487   \svgx@setformatkey{#1}{svgx@magick@set}%
1488   \FamilyKeyStateProcessed%
1489 }
1490 \newcommand*\svgx@magick@opr{}
1491 \DefineFamilyKey{SVG}{magickoperator}{%
1492   \svgx@setformatkey{#1}{svgx@magick@opr}%
1493   \FamilyKeyStateProcessed%
1494 }
1495 (/extract)

```

gsexe (opt.) Options to set the command including maybe the path to **Ghostscript**. As **Ghostscript** needs a specific device defined for different output formats, the option `gsdevice` can be used. It can either be set for all or a specific output format just like `gsopt` in the same manner like option `convertdpi`.

```

\svgx@gs@exe
gsopt (opt.)
\svgx@gs@opt
gsdevice (opt.)
\svgx@gs@device
1496 (*base)
1497 \svg@dummy@key{gsexe}
1498 \svg@dummy@key{gsopt}
1499 \svg@dummy@key{gsdevice}
1500 (/base)
1501 (*extract)
1502 \newcommand*\svgx@gs@exe{}
1503 \DefineFamilyKey{SVG}{gsexe}{%
1504   \renewcommand*\svgx@gs@exe{#1}%
1505   \FamilyKeyStateProcessed%
1506 }
1507 \newcommand*\svgx@gs@opt{}
1508 \DefineFamilyKey{SVG}{gsopt}{%
1509   \svgx@setformatkey{#1}{svgx@gs@opt}%
1510   \FamilyKeyStateProcessed%
1511 }
1512 \newcommand*\svgx@gs@device{}
1513 \DefineFamilyKey{SVG}{gsdevice}{%
1514   \svgx@setformatkey{#1}{svgx@gs@device}%
1515   \FamilyKeyStateProcessed%
1516 }
1517 (/extract)

```

C.1.3. Setting output folder

extractpath (opt.) The option `extractpath` controls, in which folder the results both of the extraction as well as the conversion of **ImageMagick** or **Ghostscript** will be located. With option `extractname` the name of the extracted and maybe converted file itself can be changed.

```

path (opt.)
extractname (opt.)
name (opt.)
\svgx@out@path
\svgx@out@name
\if@svgx@out@sec
svgx@out@count (counter)
1518 (*base)
1519 \svg@dummy@key{extractpath}

```



```

1520 \svg@dummys@key{path}
1521 \svg@dummys@key{extractname}
1522 \svg@dummys@key{name}
1523 </base>
1524 <*extract>
1525 \newcommand*{svgx@out@path}{
1526 \DefineFamilyKey{SVG}{extractpath}{%
1527 \svg@sanitize@dq\svg@tempb{#1}%
1528 \FamilySetNumerical{SVG}{extractpath}{svg@tempa}{%
1529 {svgpath}{0},{svgdir}{0},%
1530 {svgsubpath}{1},{svgsubdir}{1},%
1531 {basepath}{2},{basedir}{2},{jobpath}{2},{jobdir}{2},%
1532 {basesubpath}{3},{basesubdir}{3},{jobsubpath}{3},{jobsubdir}{3}%
1533 }{\svg@tempb}%
1534 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1535 \ifcase\svg@tempa\relax% svgpath
1536 \renewcommand*{svgx@out@path}{\svg@file@path}%
1537 \or% svgsubpath
1538 \renewcommand*{svgx@out@path}{\svg@file@path svg-extract/}%
1539 \or% basepath
1540 \renewcommand*{svgx@out@path}{./}%
1541 \or% basesubpath
1542 \renewcommand*{svgx@out@path}{./svg-extract/}%
1543 \fi%
1544 \else%
1545 \edef\svgx@out@path{\svg@tempb}%
1546 \svg@normalize@path{\svgx@out@path}%
1547 \FamilyKeyStateProcessed%
1548 \fi%
1549 }
1550 \DefineFamilyKey{SVG}{path}{%
1551 \svg@deprecated@key[svg-extract]{path=#1}{extractpath=#1}%
1552 }
1553 \newcounter{svgx@out@count}
1554 \newcommand*{svgx@out@name}{
1555 \newif\if@svgx@out@sec
1556 \DefineFamilyKey{SVG}{extractname}{%
1557 \svg@quotes@remove[#{1}]{\svg@tempb}%
1558 \FamilySetNumerical{SVG}{extractname}{svg@tempa}{%
1559 {filename}{0},{name}{0},%
1560 {filenamenumbered}{1},{namenumbered}{1},%
1561 {numberedfilename}{1},{numberedname}{1},%
1562 {numbered}{2},{section}{2},{numberedsection}{2},{sectionnumbered}{2}%
1563 }{\svg@tempb}%
1564 \@svgx@out@secfalse%
1565 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1566 \ifcase\svg@tempa\relax% filename
1567 \renewcommand*{svgx@out@name}{\svg@out@name-extract}%
1568 \or% filenamenumbered
1569 \renewcommand*{svgx@out@name}{\the\value{svgx@out@count}-\svg@out@name}%
1570 \or% numbered
1571 \renewcommand*{svgx@out@name}{\the\value{svgx@out@count}-\svgx@out@sec}%
1572 \@svgx@out@sectrue%
1573 \fi%
1574 \else%
1575 \if@svg@quotes@found%
1576 \edef\svgx@out@name{"\svg@tempb"}%
1577 \else%
1578 \edef\svgx@out@name{\svg@tempb}%
1579 \fi%
1580 \FamilyKeyStateProcessed%
1581 \fi%
1582 }
1583 \DefineFamilyKey{SVG}{name}{%
1584 \svg@deprecated@key[svg-extract]{name=#1}{extractname=#1}%
1585 }

```

C.1.4. Options for the extraction of graphics

<p>extractwidth (opt.) \svgx@param@width extractheight (opt.) \svgx@param@width extractdistort (opt.) extractkeepaspectratio (opt.) \svgx@param@distort extractscale (opt.) \svgx@param@scale</p>	<p>For graphic extraction, the given settings regarding the size for inclusion can be overwritten with these options. Using <code>\relax</code> as value leads to resetting an option as unset, regardless of what was previously given. The value <code>inherit</code> means, that the actual option for including is used for extraction as well. This is the default setting.</p> <pre> 1587 (*base) 1588 \svg@dummy@key{extractwidth} 1589 \svg@dummy@key{extractheight} 1590 \svg@dummy@key{extractdistort} 1591 \svg@dummy@key{extractkeepaspectratio} 1592 \svg@dummy@key{extractscale} 1593 </base> 1594 (*extract) 1595 \newcommand*\svgx@param@width{\svg@param@width} 1596 \DefineFamilyKey{SVG}{extractwidth}{% 1597 \FamilyKeyStateUnknownValue% 1598 \svg@ifvalueisrelax{#1}{% 1599 \renewcommand*\svgx@param@width{\z@}% 1600 \FamilyKeyStateProcessed% 1601 }{% 1602 \ifstr{#1}{inherit}{% 1603 \renewcommand*\svgx@param@width{\svg@param@width}% 1604 \FamilyKeyStateProcessed% 1605 }{% 1606 \FamilySetLengthMacro{SVG}{extractwidth}{\svg@param@width}{#1}% 1607 \ifx\FamilyKeyState\FamilyKeyStateProcessed% 1608 \ifdim\svgx@param@width<\z@\relax% 1609 \FamilyKeyStateUnknownValue% 1610 \fi% 1611 \fi% 1612 }% 1613 }% 1614 } 1615 \newcommand*\svgx@param@height{\svg@param@height} 1616 \DefineFamilyKey{SVG}{extractheight}{% 1617 \FamilyKeyStateUnknownValue% 1618 \svg@ifvalueisrelax{#1}{% 1619 \renewcommand*\svgx@param@height{\z@}% 1620 \FamilyKeyStateProcessed% 1621 }{% 1622 \ifstr{#1}{inherit}{% 1623 \renewcommand*\svgx@param@height{\svg@param@height}% 1624 \FamilyKeyStateProcessed% 1625 }{% 1626 \FamilySetLengthMacro{SVG}{extractheight}{\svg@param@height}{#1}% 1627 \ifx\FamilyKeyState\FamilyKeyStateProcessed% 1628 \ifdim\svgx@param@height<\z@\relax% 1629 \FamilyKeyStateUnknownValue% 1630 \fi% 1631 \fi% 1632 }% 1633 }% 1634 } 1635 \newif\if@svgx@param@distort 1636 \DefineFamilyKey{SVG}{extractdistort}[true]{% 1637 \FamilyKeyStateUnknownValue% 1638 \svg@ifvalueisrelax{#1}{% 1639 \@svgx@param@distortfalse% 1640 \FamilyKeyStateProcessed% 1641 }{% </pre>
---	---

```

1642 \ifstr{#1}{inherit}{%
1643 \renewcommand*\if@svgx@param@distort{\if@svg@param@distort}%
1644 \FamilyKeyStateProcessed%
1645 }{%
1646 \FamilySetBool{SVG}{extractdistort}{@svgx@param@distort}{#1}%
1647 }%
1648 }%
1649 }
1650 \DefineFamilyKey{SVG}{extractkeepaspectratio}[true]{%
1651 \FamilySetBool{SVG}{extractkeepaspectratio}{@svg@tempswa}{#1}%
1652 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1653 \if@svg@tempswa%
1654 \FamilyOptions{SVG}{extractdistort=false}%
1655 \else
1656 \FamilyOptions{SVG}{extractdistort=true}%
1657 \fi%
1658 \else%
1659 \FamilyOptions{SVG}{extractdistort=#1}%
1660 \fi%
1661 }
1662 \newcommand*\svgx@param@scale{\svg@param@scale}
1663 \DefineFamilyKey{SVG}{extractscale}{%
1664 \FamilyKeyStateUnknownValue%
1665 \svg@ifvalueisrelax{#1}{%
1666 \renewcommand*\svgx@param@scale{1}%
1667 \FamilyKeyStateProcessed%
1668 }{%
1669 \ifstr{#1}{inherit}{%
1670 \renewcommand*\svgx@param@scale{\svg@param@scale}%
1671 \FamilyKeyStateProcessed%
1672 }{%
1673 \ifisdimension{#1\p@}{%
1674 \ifdim\dimexpr#1\p@>\z@>\relax%
1675 \renewcommand*\svgx@param@scale{#1}%
1676 \FamilyKeyStateProcessed%
1677 \fi%
1678 }{%
1679 }%
1680 }%
1681 }
1682 </extract>

```

`extractpretex` (opt.) The similar hooks for executing code right before or after the graphic extraction.

```

\svgx@param@pretex
extractapptex (opt.) 1683 (*base)
\svgx@param@apptex 1684 \svg@dummys@key{extractpretex}
extractpostex (opt.) 1685 \svg@dummys@key{extractapptex}
1686 \svg@dummys@key{extractpostex}
1687 </base>
1688 (*extract)
1689 \newcommand*\svgx@param@pretex{\svg@param@pretex}
1690 \DefineFamilyKey{SVG}{extractpretex}{%
1691 \svg@ifvalueisrelax{#1}{%
1692 \let\svgx@param@pretex\relax%
1693 }{%
1694 \ifstr{#1}{inherit}{%
1695 \renewcommand*\svgx@param@pretex{\svg@param@pretex}%
1696 }{%
1697 \renewcommand*\svgx@param@pretex{#1}%
1698 }%
1699 }%
1700 \FamilyKeyStateProcessed%
1701 }
1702 \newcommand*\svgx@param@apptex{\svg@param@apptex}
1703 \DefineFamilyKey{SVG}{extractapptex}{%

```

```

1704 \svg@ifvalueisrelax{#1}{%
1705 \let\svgx@param@apptex\relax%
1706 }{%
1707 \ifstr{#1}{inherit}{%
1708 \renewcommand*\svgx@param@apptex{\svg@param@apptex}%
1709 }{%
1710 \renewcommand*\svgx@param@apptex{#1}%
1711 }%
1712 }%
1713 \FamilyKeyStateProcessed%
1714 }
1715 \DefineFamilyKey{SVG}{extractpostex}{%
1716 \svg@deprecated@key[svg-extract]{extractpostex=#1}{extractapptex=#1}%
1717 }
1718 </extract>

```

C.1.5. Miscellaneous options

`clean` (opt.) With option `clean` files generated during the extraction process can be deleted. Setting `true`
`clear` (opt.) will remove all files, `false` won't clear any file. Additionally, a specific file list of suffixes can
`\svgx@clean` be given.

```

1719 (*base)
1720 \svg@dummys@key[true]{clean}
1721 \svg@dummys@key[true]{clear}
1722 </base>
1723 (*extract)
1724 \newcommand*\svgx@clean{}
1725 \DefineFamilyKey{SVG}{clean}[true]{%
1726 \FamilySetBool{SVG}{clean}{@svg@tempswa}{#1}%
1727 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1728 \if@svg@tempswa%
1729 \renewcommand*\svgx@clean{log,aux,dvi,out,ps,eps,pdf,\svgx@latex@ext}%
1730 \else%
1731 \renewcommand*\svgx@clean{}%
1732 \fi%
1733 \else%
1734 \renewcommand*\svgx@clean{#1}%
1735 \FamilyKeyStateProcessed%
1736 \fi%
1737 }
1738 \DefineFamilyKey{SVG}{clear}{\FamilyOptions{SVG}{clean=#1}}
1739 </extract>

```

`exclude` (opt.) If it is desired not to include but only extract graphics with package `svg-extract`, option
`exclude` can be used.

```

1740 (*base)
1741 \svg@dummys@key[true]{exclude}
1742 </base>
1743 (*extract)
1744 \DefineFamilyKey{SVG}{exclude}[true]{%
1745 \FamilySetBool{SVG}{exclude}{@svg@tempswa}{#1}%
1746 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1747 \if@svg@tempswa%
1748 \renewcommand*\svg@input[2][ ]{%
1749 \if@svgx@run\else%
1750 \PackageWarning{svg-extract}{%
1751 The image ‘##2’ was\MessageBreak%
1752 neither extracted nor included%
1753 }%
1754 \fi%
1755 }%
1756 \else%

```

```

1757     \renewcommand*\svg@input{\svg@@input}%
1758     \fi%
1759     \fi%
1760 }
1761 </extract>

```

C.2. User commands

```

\includesvg The parameters angle and origin are defined as pendants to the keys provided by
extract (param.) \includegraphics.
extractpreamble (param.)
extractformat (param.) 1762 (*extract)
extractwidth (param.) 1763 \newcommand*\svgx@param@angle{0}
extractheight (param.) 1764 \svg@local@param@def{%
extractdistort (param.) 1765 \DefineFamilyKey[.param]{SVG}{extractangle}{%
extractscale (param.) 1766 \FamilyKeyStateUnknownValue%
extractangle (param.) 1767 \svg@ifvalueisrelax{#1}{%
extractpretex (param.) 1768 \renewcommand*\svgx@param@angle{0}%
extractapptex (param.) 1769 \FamilyKeyStateProcessed%
extractruns (param.) 1770 }{%
latexopt (param.) 1771 \ifstr{#1}{inherit}{%
convert (param.) 1772 \renewcommand*\svgx@param@angle{\svg@param@angle}%
convertformat (param.) 1773 \FamilyKeyStateProcessed%
convertdpi (param.) 1774 }{%
magicksetting (param.) 1775 \ifisdimension{#1\p@}{%
magickoperator (param.) 1776 \renewcommand*\svgx@param@angle{#1}%
gsopt (param.) 1777 \FamilyKeyStateProcessed%
gsdevice (param.) 1778 }{%
clean (param.) 1779 }%
exclude (param.) 1780 }%
\includeinkscape 1781 }%
extract (param.) 1782 }
extractpreamble\start Some dummies for package svg.
\svghidepreamble\end extractpreamble
extractwidth (param.) 1784 (*base)
extractheight (param.) 1785 \newcommand*\svghidepreamblestart{%
extractdistort (param.) 1786 \PackageWarning{svg}{%
extractscale (param.) 1787 The macro ‘\string\svghidepreamblestart’ is only meant\MessageBreak%
extractangle (param.) 1788 to be used together with package ‘svg-extract’.\MessageBreak%
extractpretex (param.) 1789 Nevertheless, nothing will happen%
extractapptex (param.) 1790 }%
extractruns (param.) 1791 }
latexopt (param.) 1792 \newcommand*\svghidepreambleend{%
convert (param.) 1793 \PackageWarning{svg}{%
convertformat (param.) 1794 The macro ‘\string\svghidepreambleend’ is only meant\MessageBreak%
convertdpi (param.) 1795 to be used together with package ‘svg-extract’.\MessageBreak%
magicksetting (param.) 1796 Nevertheless, nothing will happen%
magickoperator (param.) 1797 }%
gsopt (param.) 1798 }
gsdevice (param.) 1799 </base>
clean (param.)
exclude (param.)

1800 (*extract)
1801 \let\svghidepreamblestart\relax
1802 \let\svghidepreambleend\relax
1803 </extract>

```

C.3. Auxiliary macros

```

\svg@extract The macro \svg@extract does the actual job of both extracting and converting independent
\svgx@stream@in graphic files. Since it is necessary to run it with --shell-escape enabled, the command
\svgx@read@line raises a warning if it is not activated. Afterwards, the package is finished.
\svgx@stream@out
\if@svgx@preamble@write
1804 (*base)
1805 \newcommand*\svg@extract[1]{%
1806 </base>
1807 (*extract)
1808 \ifnum\pdf@shellescape=@\one\relax\else%
1809 \renewcommand*\svg@extract[1]{%
1810 \if@svgx@run%
1811 \begingroup%
1812 \edef\svg@tempa{#1}%
1813 \svg@quotes@remove{\svg@tempa}%
1814 \PackageWarning{svg-extract}{%
1815 You didn't enable 'shell escape' (or 'write18')\MessageBreak%
1816 so it wasn't possible to run the extraction for\MessageBreak%
1817 file '\svg@tempa'\MessageBreak%
1818 }%
1819 \endgroup%
1820 \fi%
1821 }%
1822 \expandafter\endinput%
1823 \fi

```

If `--shell-escape` is enabled, the command is defined with its intended functionality. Some macros and a input stream as well as a output stream are necessary for this.

```

1824 \newread\svgx@stream@in
1825 \newcommand*\svgx@read@line{}
1826 \newwrite\svgx@stream@out
1827 \newif\if@svgx@preamble@write
1828 \renewcommand*\svg@extract[1]{%

```

If option `extract` is enabled...

```

1829 \if@svgx@run%

```

...the macro `\svgx@get@out@sec` is used to get the current level numbering within the document and the counter for extracted graphics is stepped. After that, a separate auxiliary L^AT_EX file is created for extracting independent graphic files. The macro `\svgx@get@out@sec` is used to get the current level numbering within the document. The specified preamble is read for this task, if it exists. It is first searched in the same folder as the SVG file and if it wasn't found, in any other valid folder for SVG files.

```

1830 \if@svgx@out@sec%
1831 \svgx@get@out@sec%
1832 \fi%
1833 \stepcounter{svgx@out@count}%
1834 \begingroup%
1835 \def\svg@tempa##1.##2\@nil{%
1836 \IfArgIsEmpty{##2}{\edef\svgx@preamble{##1.\svgx@latex@ext}}{}}%
1837 }%
1838 \expandafter\svg@tempa\svgx@preamble.\@nil%
1839 \IfFileExists{\svg@file@path\svgx@preamble}{%
1840 \@svg@file@foundtrue%
1841 }{%
1842 \svg@get@path[]{\svgx@preamble}{\svg@out@path}%
1843 \def\svg@tempa###1.###2\@nil{%
1844 \edef\svgx@preamble{\svg@file@name.###2}%
1845 }%
1846 \expandafter\svg@tempa\svgx@preamble\@nil%
1847 }%
1848 \edef\svg@tempa{%

```

```

1849     \endgroup%
1850     \if@svg@file@found%
1851         \ifx\svg@file@path\@empty%
1852             \def\noexpand\svgx@preamble{.\svgx@preamble}%
1853         \else%
1854             \def\noexpand\svgx@preamble{\svg@file@path\svgx@preamble}%
1855         \fi%
1856     \fi%
1857 }%
1858 \svg@tempa%
1859 \begingroup%
1860     \endlinechar=\m@ne%
1861     \IfFileExists{\svgx@preamble}{%
1862         \PackageInfo{svg-extract}{%
1863             The preamble file ‘\svgx@preamble’\MessageBreak%
1864             is used for the generation of the auxiliary file\MessageBreak%
1865             ‘\svgx@out@name.\svgx@latex@ext’%
1866         }%

```

The catcodes for # need to be changed to prevent doubling when reading the line.

```

1867     \catcode‘\#=12\relax%
1868     \immediate\openout\svgx@stream@out=\svgx@out@name.\svgx@latex@ext%
1869     \immediate\openin\svgx@stream@in=\svgx@preamble%
1870     \@svg@tempswatrue%
1871     \@svgx@preamble@writetrue%
1872     \def\svgx@read@line{%

```

The given preamble file is read line by line and written to the separate auxiliary L^AT_EX file `\svgx@out@name.\svgx@latex@ext` via the output stream.

```

1873     \@whiles\if@svg@tempswa\fi{%
1874         \immediate\read\svgx@stream@in to\svgx@read@line%
1875         \ifx\svgx@read@line\@empty%
1876             \ifeof\svgx@stream@in\@svg@tempswafalse\fi%
1877         \else%

```

With `\svghidepreamblestart` and `\svghidepreambleend` it is possible for the user to omit certain parts of the preamble. Therefore the two macros `\svgx@read@preamble@till` and `\svgx@read@preamble@from` are toggling the switch `\if@svgx@preamble@write`

```

1878     \svgx@read@preamble@till{\svghidepreamblestart}{}%
1879     \svgx@read@preamble@from{\svghidepreambleend}{}%

```

If the desired end of the preamble (`\svgx@endpreamble`) was found, the readout is terminated by switching `\if@svg@tempswa` to false.

```

1880     \svgx@read@preamble@till{\svgx@endpreamble}{\@svg@tempswafalse}%
1881     \if@svgx@preamble@write%

```

During the readout process, it is searched with `\svgx@documentclass` for the appearance of `\documentclass` and `\if@svgx@classfound` is set to true if it was found.

```

1882     \if@svgx@classfound\else%
1883         \expandafter\svgx@documentclass%
1884         \svgx@read@line\documentclass\documentclass\@nil%
1885     \fi%

```

Writing out the—maybe manipulated—read in line.

```

1886     \ifx\svgx@read@line\@empty\else%
1887         \immediate\write\svgx@stream@out{%
1888             \unexpanded\expandafter{\svgx@read@line}%
1889         }%
1890     \fi%
1891     \fi%
1892 \fi%

```

```

1893     }%
1894     \immediate\closein\svgx@stream@in%
1895     \immediate\closeout\svgx@stream@out%
1896     \catcode'\#=6\relax%

```

Once the separate auxiliary L^AT_EX file is written, it is read in again and its content is stored in `\svg@tempa`, since it is necessary to prepend some stuff to the preamble, for example a maybe not existent document class.

```

1897     \immediate\openin\svgx@stream@in=\svgx@out@name.\svgx@latex@ext%
1898     \def\svg@tempa{%
1899     \loop\unless\ifeof\svgx@stream@in%
1900     \readline\svgx@stream@in to\svgx@read@line%
1901     \ifx\svgx@read@line\@empty\else%
1902     \edef\svg@tempa{%
1903     \unexpanded\expandafter{\svg@tempa}%
1904     \unexpanded\expandafter{\svgx@read@line}^^J%
1905     }%
1906     \fi%
1907     \repeat%
1908     \immediate\closein\svgx@stream@in%
1909     }{%

```

If a file was given that doesn't exist, a warning is issued.

```

1910     \svg@quotes@remove{\svgx@preamble}%
1911     \ifx\svgx@preamble\@empty\else%
1912     \PackageWarning{svg-extract}{%
1913     The preamble file '\svgx@preamble'\MessageBreak%
1914     does not exist%
1915     }%
1916     \fi%
1917     \def\svg@tempa{%
1918     }%

```

After the preamble was read in and stored in `\svg@tempa`, the separate auxiliary L^AT_EX file is written again. Some information are written right at the beginning of the file.

```

1919     \immediate\openout\svgx@stream@out=\svgx@out@name.\svgx@latex@ext%
1920     \immediate\write\svgx@stream@out{%
1921     \@percentchar\@percentchar\space This file was generated by package
1922     'svg-extract'^^J%
1923     \@percentchar\@percentchar\space from source '\jobname'^^J%
1924     \@percentchar\@percentchar\space It's intended to be compiled with
1925     '\svgx@latex@exe\ifx\svgx@latex@opt\@empty\else\space\svgx@latex@opt\fi'
1926     }%

```

With the intention of passing the correct paper dimensions, the calculating of the paper size is executed with `\AtBeginDocument` even before the document class, so that this is definitely the first thing to happen at the beginning of the document. Additionally, it is ensured that the `\special` command is definitely used with the correct paper size, when creating a DVI file.

```

1927     \immediate\write\svgx@stream@out{%
1928     \string\AtBeginDocument{\@percentchar^^J%
1929     \space\space\string\svgxsetpapersize\@percentchar^^J%
1930     \ifxetex\else\ifpdf\else%
1931     \space\space\string\AtBeginDvi{\string\special{%
1932     papersize=\string\the\string\paperwidth,%
1933     \string\the\string\paperheight%
1934     }}\@percentchar^^J%
1935     \fi\fi%
1936     }^^J%
1937     \string\PassOptionsToPackage{hidelinks}{hyperref}%
1938     }%

```


If no document class was found during reading the preamble file, then class `\article` is used.

```

1939     \if@svgx@classfound\else%
1940         \immediate\write\svgx@stream@out{\string\documentclass{article}}%
1941     \fi%

```

And now the stored preamble.

```

1942     \ifx\svg@tempa\@empty\else%
1943         \immediate\write\svgx@stream@out{\unexpanded\expandafter{\svg@tempa}}%
1944     \fi%

```

After the given preamble was written, package **svg-extract** will be loaded in case it was forgotten.

```

1945     \immediate\write\svgx@stream@out{\string\usepackage{svg-extract}}%

```

Now all parameters relevant for the extraction are evaluated and appended.

```

1946     \def\svg@tempa##1{%
1947         \immediate\write\svgx@stream@out{\string\svgsetup{##1}}%
1948     }%
1949     \if@svg@ink@latex\else%
1950         \svg@tempa{inkscapelatex=false}%
1951     \fi%
1952     \ifdim\svgx@param@width>\z@\relax%
1953         \svg@tempa{width=\svgx@param@width}%
1954     \fi%
1955     \ifdim\svgx@param@height>\z@\relax%
1956         \svg@tempa{height=\svgx@param@height}%
1957     \fi%
1958     \if@svgx@param@distort%
1959         \svg@tempa{distort=true}%
1960     \fi%
1961     \ifdim\dimexpr\svgx@param@scale\p@\relax=\p@\relax\else%
1962         \svg@tempa{scale=\svgx@param@scale}%
1963     \fi%
1964     \def\svg@tempb{\svg@param@pretex}%
1965     \ifx\svgx@param@pretex\svg@tempb\relax%
1966         \let\svgx@param@pretex\svg@param@pretex%
1967     \fi%
1968     \ifx\svgx@param@pretex\relax\else%
1969         \svg@tempa{pretex=\unexpanded\expandafter{\svgx@param@pretex}}%
1970     \fi%
1971     \def\svg@tempb{\svg@param@apptex}%
1972     \ifx\svgx@param@apptex\svg@tempb\relax%
1973         \let\svgx@param@apptex\svg@param@apptex%
1974     \fi%
1975     \ifx\svgx@param@apptex\relax\else%
1976         \svg@tempa{apptex=\unexpanded\expandafter{\svgx@param@apptex}}%
1977     \fi%

```

Parameter `lastpage` is only considered for including PDF files with L^AT_EX support.

```

1978     \let\svg@tempa\@empty%
1979     \if@svg@ink@latex%
1980         \ifstr{\svg@ink@format}{pdf}{%
1981             \ifnum\value{svg@param@lastpage}>\z@\relax%
1982                 \edef\svg@tempa{lastpage=\the\value{svg@param@lastpage}}%
1983             \else%
1984                 \ifnum\value{svg@param@lastpage}=\z@\relax%
1985                     \def\svg@tempa{lastpage=true}%
1986                 \else%
1987                     \def\svg@tempa{lastpage=false}%
1988                 \fi%
1989             \fi%

```

```

1990     }{}%
1991     \fi%

```

The rotation angle, if given.

```

1992     \ifdim\dimexpr\svgx@param@angle\p@\relax=\z@\relax\else%
1993     \edef\svg@tempa{%
1994         angle=\svgx@param@angle\ifx\svg@tempa\@empty\else,\svg@tempa\fi%
1995     }%
1996     \fi%

```

As we are now at the end of the preamble and just before the beginning of the document, the paper dimension are set again to make sure, that these settings are active at the end of the preamble. Additionally, it is executed again at the very end of `\AtBeginDocument` to ensure, that no other package used this hook for manipulating the paper size.

```

1997     \ifx\svg@tempa\@empty%
1998     \def\svg@tempa{\string\svgxsetbox{#1}}%
1999     \else%
2000     \edef\svg@tempa{\noexpand\string\noexpand\svgxsetbox[\svg@tempa]{#1}}%
2001     \fi%
2002     \immediate\write\svgx@stream@out{\svg@tempa}%

```

Package `xr` is used to evaluate possible labels within the included *Inkscape* L^AT_EX file.

```

2003     \if@svg@ink@latex%
2004     \IfFileExists{xr.sty}{%
2005         \immediate\write\svgx@stream@out{%
2006             \string\usepackage{xr}^^J%
2007             \string\externaldocument{\jobname}^^J%
2008         }%
2009     }{}%
2010     \fi%
2011     \immediate\write\svgx@stream@out{%
2012         \string\begin{document}^^J%
2013         \string\pagestyle{empty}^^J%
2014         \string\svgxoutputbox\@percentchar^^J%
2015         \string\end{document}%
2016     }%
2017     \immediate\closeout\svgx@stream@out%
2018     \endgroup%

```

After creating the separate auxiliary L^AT_EX file, the actual extraction and conversion can be done.

```

2019     \ifstr{\svgx@format\svgx@cnv@format}{-}{%
2020         \PackageWarning{svg-extract}{%
2021             Both keys ‘extractformat’ and ‘convertformat’ are\MessageBreak%
2022             empty, so nothing to do so far%
2023         }%
2024     }{}%

```

As the extraction maybe needs to include the main auxiliary file with `\externaldocument` provided by package `xr` it is necessary to do all related stuff after the main auxiliary file was written. This is done with `\AfterReadingMainAux` provided by package `scrfile`.

```

2025     \svg@quotes@remove{\svgx@out@path}%
2026     \svg@quotes@remove{\svgx@out@name}%

```

All generated files will be moved to the desired output folder, which is given by option `extractpath`. Therefore, this folder is created.

```

2027     \edef\svg@tempb{%
2028         \noexpand\svg@shell@mkdir{\svgx@out@path}%
2029     }%
2030     \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%

```

First of all the separate auxiliary L^AT_EX file is compiled with the detected L^AT_EX processor (`\svgx@latex@exe`) as often as defined by counter option `extractruns`.

```

2031     \edef\svg@tempb{%
2032         \noexpand\PackageInfo{svg-extract}{%
2033             Running LaTeX (\svgx@latex@exe) for graphic extraction%
2034             \ifx\svgx@latex@opt\@empty\else%
2035                 \MessageBreak with added options ‘\svgx@latex@opt’%
2036             \fi%
2037         }%
2038     }%
2039     \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2040     \edef\svg@tempb{%
2041         \noexpand\ShellEscape{%
2042             \svgx@latex@exe\space\svgx@latex@opt\space%
2043             "\svgx@out@name.\svgx@latex@ext"%
2044         }%
2045     }%
2046     \loop\ifnum\value{svgx@runs}>\z@\relax%
2047         \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2048         \advance\c@svgx@runs\m@ne%
2049     \repeat%

```

All files requested with option `extractformat` are created with internal conversion tools supplied by most L^AT_EX 2_ε distributions if necessary.

```

2050     \def\svg@tempa##1##2##3{%
2051         \edef\svg@tempb{%
2052             \noexpand\ShellEscape{%
2053                 \@nameuse{svgx@##1@exe}\space\@nameuse{svgx@##1@opt}\space%
2054                 "\svgx@out@name.##2"%
2055             }%
2056         }%
2057         \AfterReadingMainAux{\PackageInfo{svg-extract}{Running ##1}}%
2058         \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2059     }%
2060     \@svg@tempswafalse%
2061     \ifxetex\else\ifpdf\else%
2062         \@svg@tempswatruetrue%
2063     \fi\fi%
2064     \if@svg@tempswa%
2065         \svg@tempa{dvips}{dvi}{ps}%
2066         \svgx@ifinlist{eps}{\svgx@format}{\svg@tempa{pstoeeps}{ps}{eps}}{}%
2067         \svgx@ifinlist{pdf}{\svgx@format}{\svg@tempa{pstopdf}{ps}{pdf}}{}%
2068     \else%
2069         \svgx@ifinlist{eps}{\svgx@format}{\svg@tempa{pdftoeeps}{pdf}{eps}}{}%
2070         \svgx@ifinlist{ps}{\svgx@format}{\svg@tempa{pdftops}{pdf}{ps}}{}%
2071     \fi%

```

Now the desired conversion tool is invoked if requested.

```

2072     \if@svgx@cnv@run%

```

If no density was given at all, the density for PNG files is set to 300dpi by default.

```

2073         \ifx\svgx@cnv@dpi\relax%
2074             \ifx\svgx@cnv@dpi@png\@undefined%
2075                 \def\svgx@cnv@dpi@png{300}%
2076             \fi%
2077         \fi%

```

The first given file type with option `extractformat` is used as source for the conversion process.

```

2078         \expandafter\svgx@cnv@get@informat\expandafter{\svgx@format}%

```

The conversion is done for each desired file type given in a list by option `convertformat`.

```

2079     \@for\svg@tempa:=\svgx@cnv@format\do{%
2080     \ifx\svg@tempa\@empty\else%
2081     \expandafter\svgx@ifinlist\expandafter{\svg@tempa}{\svgx@format}{%
2082     \PackageWarning{svg-extract}{%
2083     File type ‘\svg@tempa’ was specified for option\MessageBreak%
2084     ‘extractformat’ (\svgx@format) as well as for \MessageBreak%
2085     option ‘convertformat’ (\svgx@cnv@format) so the\MessageBreak%
2086     conversion won’t be done%
2087     }%
2088     }{%
2089     \edef\svg@tempb{%
2090     \noexpand\PackageInfo{svg-extract}{%
2091     Converting ‘\svgx@out@name.\svgx@cnv@informat’\MessageBreak%
2092     to ‘\svgx@out@name.\svg@tempa’%
2093     }%
2094     }%
2095     \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2096     \edef\svg@tempb{%
2097     \noexpand\ShellEscape{%
2098     \svgx@cnv@cmd{\svgx@out@name}{\svgx@cnv@informat}{\svg@tempa}%
2099     }%
2100     }%
2101     \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2102     }%
2103     \fi%
2104     }%
2105     \fi%

```

As both extraction and conversion are done, all files are moved to the desired output folder (`extractpath`).

```

2106     \edef\svg@tempa{\svgx@format\if\svgx@cnv@run,\svgx@cnv@format\fi}%
2107     \@for\svg@tempb:=\svg@tempa\do{%
2108     \ifx\svg@tempb\@empty\else%
2109     \edef\svg@tempb{%
2110     \noexpand\svgx@move{\svgx@out@name}{\svg@tempb}{\svgx@out@path}%
2111     }%
2112     \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2113     \fi%
2114     }%

```

At the very end, all unwanted auxiliary files are deleted.

```

2115     \@for\svg@tempa:=\svgx@clean\do{%
2116     \expandafter\svgx@ifinlist\expandafter{\svg@tempa}{\svg@tempb}{%
2117     \edef\svg@tempb{%
2118     \noexpand\IfFileExists{"\svgx@out@name" . \svg@tempa}{%
2119     \noexpand\svg@shell@rm{\svgx@out@name . \svg@tempa}%
2120     }{}%
2121     }%
2122     \expandafter\AtEndDocument\expandafter{%
2123     \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2124     }%
2125     }%
2126     }%
2127     }%
2128     \fi%
2129 }
2130 </extract>

```

`\svgx@get@out@sec` The macro `\svgx@get@out@sec` reads all sectioning counters in order to get the numbering of the current sectioning level. The value is stored in `\svgx@out@sec`.

```

2131 \newcommand*\svgx@out@sec{unknown}

```

```

2132 \newcommand*{svgx@get@out@sec}{%
2133   \begingroup%
2134   \def{svg@tempa}{%
2135     \@for{svg@tempb}={%
2136       part,chapter,section,subsection,subsubsection,paragraph,subparagraph%
2137     }\do{%
2138       \ifx{svg@tempb}\@empty\else%
2139         \scr@ifundefinedorrelax{the{svg@tempb}}{-%
2140           \ifnum\value{svg@tempb}>\z@\relax%
2141             \edef{svg@tempa}{svg@tempb}%
2142             \fi%
2143           }%
2144         \fi%
2145       }%
2146     \edef{svg@tempb}{%
2147       \endgroup%
2148       \ifx{svg@tempa}\@empty\else%
2149         \def\noexpand{svgx@out@sec}{\csname the{svg@tempa}\endcsname}%
2150         \fi%
2151       }%
2152   }{svg@tempb}%
2153 }

```

`\svgx@documentclass` This delimited macro is used to find a occurrence of `\documentclass` within a read in line.
`\if@svgx@classfound` The delimiter `\documentclass` is used twice in order to ignore the possible occurrence of white space or anything else right before `\documentclass`.

```

2154 \newif\if@svgx@classfound
2155 \newcommand*{svgx@documentclass}{%
2156   \def{svgx@documentclass#1\documentclass#2\documentclass#3\@nil}{%
2157     \IfArgIsEmpty{#2}{}\@svgx@classfoundtrue}%
2158 }

```

`\svgx@read@preamble@till` These macros are used to skip some parts of a read in preamble file.
`\svgx@read@preamble@from`
`\svgx@read@preamble@skip`

```

2159 \newcommand*{svgx@read@preamble@till}[2]{%
2160   \svgx@read@preamble@skip#1\@nil{till}{#2}%
2161 }
2162 \newcommand*{svgx@read@preamble@from}[2]{%
2163   \svgx@read@preamble@skip#1\@nil{from}{#2}%
2164 }

```

In principle, the functionality is the same as for `\svgx@documentclass`.

```

2165 \newcommand*{svgx@read@preamble@skip}{%
2166   \def{svgx@read@preamble@skip#1\@nil#2#3}{%

```

A given token is used to create the macro `\svg@tempa` delimited by the token itself which is used twice to get any stuff right before or after the occurrence.

```

2167   \def{svg@tempa##1}{%
2168     \def{svg@tempa###1##1###2##1###3\@nil}{%
2169       \IfArgIsEmpty{###3}{-%

```

Write everything which was found right before the macro which starts hiding area to the output stream and stop writing with `\if@svgx@preamble@write`.

```

2170         \ifstr{#2}{till}{-%
2171           \IfArgIsEmpty{###1}{-%
2172             \immediate\write\svgx@stream@out{###1}%
2173           }%
2174         \@svgx@preamble@writefalse%
2175       }-%

```

Write everything which was found right after the macro which ends the hiding area and start writing again with `\if@svgx@preamble@write`.

```

2176     \ifstr{#2}{from}{%
2177         \IfArgIsEmpty{###2}{%
2178             \def\svgx@read@line{}}%
2179         }{%
2180             \def\svgx@read@line{###2}%
2181         }%
2182     \@svgx@preamble@writetrue%
2183 }{%
2184 }%

```

Additional stuff which should be done.

```

2185     #3%
2186 }%
2187 }%
2188 }%

```

Creating the macro `\svg@tempa` delimited by the first argument.

```

2189 \edef\svg@tempb{\expandafter\detokenize\expandafter{#1}}%
2190 \expandafter\svg@tempa\expandafter{\svg@tempb}%

```

Calling the created macro.

```

2191 \edef\svg@tempb{%
2192     \expandafter\detokenize\expandafter{\svgx@read@line}\svg@tempb\svg@tempb%
2193 }%
2194 \expandafter\svg@tempa\svg@tempb\@nil%
2195 }

```

`\svgx@cnv@informat` The first list entry from argument (`\svgx@format`) is extracted by `\svgx@cnv@get@informat`.
`\svgx@cnv@get@informat`

```

2196 \newcommand*\svgx@cnv@informat{}
2197 \newcommand*\svgx@cnv@get@informat[1]{%
2198     \begingroup%
2199     \def\svg@tempa##1,##2\@nil{%
2200         \def\svg@tempa{##1}}%
2201     }%
2202     \svg@tempa#1,\@nil%
2203 \edef\svg@tempa{%
2204     \endgroup%
2205     \def\noexpand\svgx@cnv@informat{\svg@tempa}%
2206 }%
2207 \svg@tempa%

```

If the first argument (`\svgx@format`) was empty, `\svgx@cnv@informat` is set to the a file type, which is generated anyway.

```

2208 \ifx\svgx@cnv@informat\@empty%
2209     \renewcommand*\svgx@cnv@informat{pdf}%
2210 \ifxetex\else\ifpdf\else%
2211     \renewcommand*\svgx@cnv@informat{ps}%
2212 \fi\fi%
2213 \fi%
2214 }

```

`\svgx@magick@cmd` Depending on option `convert`, one of these two macros is actually used by `\svgx@cnv@cmd`.
`\svgx@gs@cmd` For invoking the conversion process, the required platform-dependent executable is set, if nothing was set by a package option.

```

2215 \ifx\svgx@magick@exe\@empty
2216     \ifwindows
2217         \renewcommand*\svgx@magick@exe{magick}

```

```

2218 \else
2219 \renewcommand*\svgx@magick@exe{convert}
2220 \fi
2221 \fi
2222 \newcommand*\svgx@magick@cmd[3]{%
2223 \svgx@magick@exe\space%
2224 \svgx@useformatkey{svgx@cnv@dpi}{#3}{-density }%
2225 \svgx@useformatkey{svgx@magick@set}{#3}{}%
2226 "#1.#2"\space%
2227 \svgx@useformatkey{svgx@magick@opr}{#3}{}%
2228 "#1.#3"%
2229 }

2230 \ifx\svgx@gs@exe\@empty
2231 \ifwindows
2232 \renewcommand*\svgx@gs@exe{gswin64c}
2233 \else
2234 \renewcommand*\svgx@gs@exe{gs}
2235 \fi
2236 \fi
2237 \newcommand*\svgx@gs@cmd[3]{%
2238 \svgx@gs@exe\space-dSAFER -dBATC -dNOPAUSE\space%
2239 \svgx@useformatkey{svgx@gs@device}{#3}{-sDEVICE=%}%
2240 \svgx@useformatkey{svgx@cnv@dpi}{#3}{-r}%
2241 \svgx@useformatkey{svgx@gs@opt}{#3}{}%
2242 -sOutputFile="#1.#3"\space"#1.#2"%
2243 }

```

\svgx@move If the file doesn't exist

```

2244 \newcommand*\svgx@move[3]{%
2245 \begingroup%
2246 \IfFileExists{"#1" .#2}{%
2247 \svg@shell@move{#1.#2}{#3#1.#2}%
2248 }{%
2249 \edef\svg@tempa{#2}%
2250 \@svg@tempswafalse%
2251 \expandafter\svgx@ifinlist\expandafter{\svg@tempa}{\svgx@cnv@format}{%
2252 \@svg@tempswatrue%
2253 \def\svg@tempb{conversion}%
2254 }{%
2255 \expandafter\svgx@ifinlist\expandafter{\svg@tempa}{pdf,ps,eps}{%
2256 \@svg@tempswatrue%
2257 \def\svg@tempb{extraction}%
2258 }{}%
2259 }%
2260 \if@svg@tempswa%
2261 \edef\svg@tempb{%
2262 The graphic file \svg@tempb\space failed\MessageBreak%
2263 for '#1.#2'\MessageBreak%
2264 Troubleshooting: Please check the log file how\MessageBreak%
2265 the invocation of the extraction took place and\MessageBreak%
2266 try to execute it yourself in the terminal%
2267 }%
2268 \else%
2269 \def\svg@tempb{%
2270 The extraction to format '#2' failed\MessageBreak%
2271 for '#1.#2'\MessageBreak%
2272 Only file types 'pdf,ps,eps' are supported for\MessageBreak%
2273 key 'exportformat'%
2274 }%
2275 \fi%
2276 \PackageWarning{svg-extract}{\svg@tempb}%
2277 }%
2278 \endgroup%
2279 }

```

`\svgx@ifinlist` Check, if the first argument is included in a comma-separated list in the second argument. Keep in mind that the first argument is not expanded at all, the second one exactly once.

```
2280 \newcommand*\svgx@ifinlist[2]{%
2281   \begingroup%
2282   \def\svg@tempa##1,##2\@nil{%
2283     \IfArgIsEmpty{##2}{%
2284       \aftergroup\@secondoftwo%
2285     }{%
2286       \aftergroup\@firstoftwo%
2287     }%
2288   }%
2289   \expandafter\svg@tempa\expandafter,##2,##1,\@nil%
2290 \endgroup%
2291 }
```

`\svgx@onlywindows` Do only some stuff, if Windows was detected.

```
2292 \newcommand*\svgx@onlywindows[1]{%
2293 \AfterPackage*{ifplatform}{\renewcommand*\svgx@onlywindows[1]{\ifwindows#1\fi}}
```

`\svgx@ifkeyandval` It is checked whether a key was given as `<key>=<value>` or like `<key>={<format>=<value>}`.

```
2294 \newcommand*\svgx@ifkeyandval[3]{%
2295   \def\svg@tempa##1==##2==##3\@nil{\ifstr{##3}{=}##2}{##3}}%
2296   \svg@tempa#1==\@nil%
2297 }
```

`\svgx@cnv@get@dpi` This macro is used to resolve a given value to set the density for the conversion. The delimited macros `\svg@tempa` and `\svg@tempb` are defined to first crop any given suffix `dpi` and second to split two numbers at `x`, if present. Pay attention how both macros are invoked. In the end, a passed value in any of the forms `300`, `300dpi`, `300x400` or `300x400dpi` and even `300dpix400dpi` is possible. The result is stored in `\svg@tempa`.

```
2298 \newcommand*\svgx@cnv@get@dpi[1]{%
2299   \begingroup%
2300   \def\svg@tempa##1dpi##2x##3dpi##4\@nil{%
2301     \edef\svg@tempa{##1}%
```

Switch `\if@svg@tempswa` as `\iftrue` means, a valid value was found.

```
2302   \@svg@tempswafalse%
```

If only the first argument is a number and third is empty, a single number was given and there's nothing more to do. If the argument is something like `300dpix400dpi`, the third argument is the second number.

```
2303   \ifnumber{##1}{%
2304     \IfArgIsEmpty{##3}{\@svg@tempswatrue}{%
2305       \ifnumber{##3}{\edef\svg@tempa{##1x##3}}}%
2306   }%
2307 }{%
2308   \if@svg@tempswa\else%
2309     \expandafter\svg@tempb\svg@tempa xx\@nil%
2310   \fi%
2311 }%
```

Macro `\svg@tempb` splits at `x` and checks, if something valid like `300x400` was given. If true, the value is stored in `\svg@tempa`.

```
2312   \def\svg@tempb##1x##2x##3\@nil{%
2313     \ifstr{##3}{x}{%
2314       \@svg@tempswatrue%
2315     \IfArgIsEmpty{##1}{\@svg@tempswafalse}{%
2316       \ifnumber{##1}{\@svg@tempswafalse}%
```



```

2317     }%
2318     \IfArgIsEmpty{##2}{\@svg@tempwafalse}{%
2319     \ifnumber{##2}{\@svg@tempwafalse}%
2320     }%
2321     \if@svg@tempswa%
2322     \edef\svg@tempa{##1x##2}%
2323     \fi%
2324   }{}%
2325 }%
2326 \IfArgIsEmpty{#1}{%
2327   \let\svg@tempa\@empty%
2328 }{%
2329   \lowercase{\svg@tempa#1dpi#1xdpi\@nil}%
2330   \if@svg@tempswa\else%
2331     \let\svg@tempa\relax%
2332   \fi%
2333 }%
2334 \edef\svg@tempb{%
2335   \endgroup%
2336   \ifx\svg@tempa\relax%
2337     \let\noexpand\svg@tempa\noexpand\relax%
2338   \else%
2339     \def\noexpand\svg@tempa{\svg@tempa}%
2340   \fi%
2341 }%
2342 \svg@tempb%
2343 }

```

`\svgx@setformatkey` With `\svgx@setformatkey` the—maybe output format depend—keys for the conversion tools are set. First argument contains the value given to a key, second the command sequence name of the macro, to whom the value shall be allocated.

```
2344 \newcommand*\svgx@setformatkey[2]{%
```

A key of the form $\langle key \rangle = \{ \langle format \rangle = \langle value \rangle \}$ is given. The desired output format can be accessed with `##1`, the value with `##2` within the arguments of `\svgx@ifkeyandval`.

```

2345 \svgx@ifkeyandval{#1}{%
2346   \svg@ifvalueisrelax{##2}{%
2347     \expandafter\let\csname #2@##1\endcsname\relax%
2348   }{%
2349     \@namedef{#2@##1}{##2}%
2350   }%

```

A key of the form $\langle key \rangle = \{ \langle format \rangle = \langle value \rangle \}$ is given. The value can be used with `##1`.

```

2351 }{%
2352   \svg@ifvalueisrelax{##1}{%
2353     \expandafter\let\csname #2\endcsname\relax%
2354   }{%
2355     \@namedef{#2}{##1}%
2356   }%
2357 }%
2358 }

```

The command `\svgx@useformatkey` checks, if a format specific key was defined with `\svgx@setformatkey`, whereas the format is given in the second argument. If this is not the case, the setting for all output formats is used. After that, a specific key appended with a `+` can be used to do some additional stuff.

```

2359 \newcommand*\svgx@useformatkey[3]{%
2360   \scr@ifundefinedorrelax{#1@#2}{%
2361     \scr@ifundefinedorrelax{#1}{}%
2362     \expandafter\ifx\csname #1\endcsname\@empty\else%
2363       #3\@nameuse{#1}\space%
2364     \fi%

```

```

2365 }%
2366 \scr@ifundefinedorrelax{#1@#2+}{-}{%
2367 \expandafter\ifx\csname #1@#2+\endcsname\@empty\else%
2368 #3\@nameuse{#1@#2+}\space%
2369 \fi%
2370 }%
2371 }{-}%

```

If this a format specific key was defined, it is used.

```

2372 \expandafter\ifx\csname #1@#2\endcsname\@empty\else%
2373 #3\@nameuse{#1@#2}\space%
2374 \fi%
2375 }%
2376 }

```

C.4. Commands for the separate auxiliary L^AT_EX-file

For the extraction of independent graphics, an auxiliary L^AT_EX file is needed. Within this file, the following commands are used to include the desired graphic.

<code>\svgxsetbox</code>	Within the preamble of the auxiliary L ^A T _E X file, the desired graphic is used to setup a box,
<code>\svgx@setbox</code>	which is used both to define the papersize as well as for the output itself. The macro
<code>\if@svgx@standalone</code>	<code>\svgx@setbox</code> is executed twice, the first time in the preamble and the second time at the
	very end of <code>\AtBeginDocument</code> if package etoolbox was loaded.

The switch `\if@svgx@standalone` is defined for enabling classes to implement a different behaviour for **svg-extract** in standalone mode. for example, TUD-Script-classes are using this switch.

```

2377 \newif\if@svgx@standalone
2378 \newcommand*\svgxsetbox[2] []{%
2379 \@svgx@standalonetrue%
2380 \svgx@setbox{#1}{#2}%
2381 \scr@ifundefinedorrelax{AtEndPreamble}{-}{%
2382 \let\svg@tempa\@firstofone%
2383 }{-}%
2384 \def\svg@tempa{\AtEndPreamble}%
2385 }%
2386 \svg@tempa{\AtBeginDocument{\svgx@setbox{#1}{#2}}}%
2387 }
2388 \newcommand*\svgx@setbox[2]{%
2389 \sbox\svg@box{\svg@input[#{#1},draft=false]{#2}}%
2390 \svgxsetpapersize%
2391 }

```

<code>\svgxsetpapersize</code>	This macro sets all well known length macros for defining the paper size as well as the type area to the size of <code>\svg@box</code> .
--------------------------------	--

```

2392 \newcommand*\svgxsetpapersize{%
2393 \setlength\paperwidth{\the\wd\svg@box}%

```

Due to the fact, that the lengths for stock- and mediasizes are maybe set to `\relax`, these macros are checked with `\scr@ifundefinedorrelax`.

```

2394 \scr@ifundefinedorrelax{stockwidth}{-}{%
2395 \setlength\stockwidth{\paperwidth}%
2396 }%
2397 \scr@ifundefinedorrelax{mediawidth}{-}{%
2398 \setlength\mediawidth{\paperwidth}%
2399 }%
2400 \setlength\textwidth{\paperwidth}%
2401 \setlength\paperheight{\the\dimexpr\ht\svg@box+\dp\svg@box\relax}%
2402 \scr@ifundefinedorrelax{stockheight}{-}{%

```

```

2403 \setlength\stockheight{\paperheight}%
2404 }%
2405 \scr@ifundefinedorrelax{mediaheight}{-}{-%
2406 \setlength\mediaheight{\paperheight}%
2407 }%
2408 \setlength\textheight{\paperheight}%

```

Any other length regarding the layout is set to have no influence at all. Hence the document has the same size as the graphic.

```

2409 \hoffset=-1in%
2410 \oddsidemargin=\z@%
2411 \evensidemargin=\z@%
2412 \voffset=-1in%
2413 \topmargin=\z@%
2414 \headheight=\z@%
2415 \headsep=\z@%
2416 \topskip=\z@%
2417 \footskip=\z@%
2418 \marginparsep=\z@%
2419 \marginparwidth=\z@%
2420 \marginparpush=\z@%
2421 }
2422 \@onlypreamble\svgxsetpapersize

```

`\svgxoutputbox` With `\svgxoutputbox` the created box is displayed.
`\if@svgx@beamer`

```

2423 \newif\if@svgx@beamer
2424 \@ifclassloaded{beamer}{\@svgx@beamertrue}{-}{-%
2425 \newcommand*\svgxoutputbox{%
2426 \begingroup%
2427 \setlength\parindent{\z@}%
2428 \setlength\parskip{\z@}%
2429 \setlength\parfillskip{\z@}%
2430 \if@svgx@beamer%
2431 \setbeamertemplate{navigation symbols}{-}{-%
2432 \begin{frame}[plain]%
2433 \usebox\svg@box%
2434 \end{frame}%
2435 \else%
2436 \usebox\svg@box%
2437 \fi%
2438 \endgraf%
2439 \endgroup%
2440 }

```

D. Processing Options

Setting the default options and processing the given ones during when loading the packages.

```

2441 (*base)
2442 \FamilyExecuteOptions{SVG}{-%
2443 inkscape=true,inkscapepath=basesubdir,
2444 inkscapelatex=true,inkscapearea=drawing,distort=false,%
2445 usexcolor=true,usetransparent=true%
2446 }
2447 </base)
2448 (*extract)
2449 \FamilyExecuteOptions{SVG}{-%
2450 extract=true,extractpath=basesubdir,%
2451 extractruns=2,extractname=namenumbered,extractdistort=false,%
2452 convert=magick,convert=false,%
2453 gsdevice={png=png16m},gsdevice={jpeg=jpeg},gsdevice={jpg=jpeg},%
2454 gsdevice={tif=tiff48nc},gsdevice={tiff=tiff48nc},%

```

```

2455 gsdevice={eps=eps2write},gsdevice={ps=ps2write}%
2456 }
2457 </extract>
2458 \FamilyProcessOptions{SVG}

```

E. Macros for file access

Finally, platform dependend macros for creating directories as well as moving and deleting files are provided, if `--shell-escape` is enabled. Only then package **ifplatform** is only used in order to do not raise a warning.

```

2459 \ifnum\pdf@shellescape=\@ne\relax\else%
2460 \expandafter\endinput%
2461 \fi
2462 \RequirePackage{ifplatform}[2010/10/22]

```

```

\svg@shell@mkdir The platform dependent commands for file access.
\svg@shell@mkdir
  \svg@shell@mv 2463 \ifwindows
  \svg@shell@mv 2464 \newcommand*\svg@shell@mmkdir[1]{if not exist "#1" mkdir "#1"}
  \svg@shell@rm 2465 \newcommand*\svg@shell@mmove{move}
  \svg@shell@rm 2466 \newcommand*\svg@shell@mrm{del}
  \svg@shell@rm 2467 \else
  \svg@shell@rm 2468 \newcommand*\svg@shell@mmkdir[1]{mkdir -p "#1"}
  \svg@shell@rm 2469 \newcommand*\svg@shell@mmove{mv}
  \svg@shell@rm 2470 \newcommand*\svg@shell@mrm{rm}
  \svg@shell@rm 2471 \fi

```

A directory should only be created, if it isn't the current working directory.

```

2472 \newcommand*\svg@shell@mmkdir[1]{%
2473 \begingroup%
2474 \svg@quotes@remove[#{1}]{\svg@tempa}%
2475 \@svg@tempswatruer%
2476 \ifstr{\svg@tempa}{.}{\@svg@tempswafalse}{%
2477 \ifstr{\svg@tempa}{.}{\@svg@tempswafalse}{%
2478 }}%
2479 \if@svg@tempswa%
2480 \ShellEscape{\svg@shell@mmkdir{\svg@tempa}}%
2481 \fi%
2482 \endgroup%
2483 }

```

Commands for moving and deleting files.

```

2484 \newcommand*\svg@shell@mmove[2]{%
2485 \ShellEscape{\svg@shell@mmove\space"#1"\space"#2"}%
2486 }
2487 \newcommand*\svg@shell@mrm[1]{%
2488 \ShellEscape{\svg@shell@mrm\space"#1"}%
2489 }

```

At the very end, the catcodes are restored.

```

2490 \svg@catcodecodes@restore

```

Index

Numbers written in *italic* refer to the page where the corresponding entry is described. Numbers underlined refer to the code line of the definition.

A	
apptex (opt.)	5, 311
C	
clean (opt.)	9, 1719
clear (opt.)	1719
convert (opt.)	10, 1346
convertdensity (opt.)	1449
convertdpi (opt.)	10, 1449
convertformat (opt.)	10, 1422
counters:	
svg@param@lastpage	334
svgx@out@count	1518
svgx@runs	1242
D	
distort (opt.)	5, 254
draft (opt.)	6, 348
dvipsopt (opt.)	9, 1297
E	
end (opt.)	1215
eps (opt.)	1174
exclude (opt.)	9, 1740
ext (opt.)	216
extension (opt.)	216
extract (opt.)	8, 1142
extractapptex (opt.)	8, 1683
extractdistort (opt.)	8, 1587
extractformat (opt.)	8, 1174
extractheight (opt.)	8, 1587
extractkeepaspectratio (opt.)	1587
extractname (opt.)	8, 1518
extractpath (opt.)	8, 1518
extractpostex (opt.)	1683
extractpreamble (opt.)	8, 1215
extractpreambleend (opt.)	8, 1215
extractpretex (opt.)	8, 1683
extractruns (opt.)	9, 1242
extractscale (opt.)	8, 1587
extractwidth (opt.)	8, 1587
G	
gsdevice (opt.)	11, 1496
gsexex (opt.)	11, 1496
gsopt (opt.)	11, 1496
H	
height (opt.)	5, 254
I	
\if@svg@draft	348
\if@svg@file@found	489
\if@svg@ink@run	849
\if@svg@param@distort	254
\if@svg@quotes@found	387
\if@svg@tempswa	14
\if@svg@use@transparent	32
\if@svg@use@xcolor	32
\if@svgx@beamer	2423
\if@svgx@classfound	2154
\if@svgx@cnv@run	1346
\if@svgx@out@sec	1518
\if@svgx@preamble@write	1804
\if@svgx@run	1142
\if@svgx@standalone	2377
\includeinkscape	6–7, 766 , 1762
angle (param.)	6 , 807
apptex (param.)	6 , 807
clean (param.)	7, 1762
convert (param.)	7, 1762
convertdpi (param.)	7, 1762
convertformat (param.)	7, 1762
distort (param.)	6, 807
draft (param.)	6, 807
exclude (param.)	7, 1762
extract (param.)	7, 1762
extractangle (param.)	7, 1762
extractapptex (param.)	7, 1762
extractdistort (param.)	7, 1762
extractformat (param.)	7, 1762
extractheight (param.)	7, 1762
extractpreamble (param.)	7, 1762
extractpretex (param.)	7, 1762
extractruns (param.)	7, 1762
extractscale (param.)	7, 1762
extractwidth (param.)	7, 1762
gsdevice (param.)	7, 1762
gsopt (param.)	7, 1762
height (param.)	6, 807
inkscapeformat (param.)	6, 807
inkscapelatex (param.)	6, 807
lastpage (param.)	6, 807
latexopt (param.)	7, 1762
magickoperator (param.)	7, 1762
magicksetting (param.)	7, 1762
origin (param.)	6, 807
pretex (param.)	6, 807
scale (param.)	6, 807
width (param.)	6, 807
\includesvg	6–7, 715 , 1762
angle (param.)	6, 750
apptex (param.)	6, 718
clean (param.)	7, 1762
convert (param.)	7, 1762
convertdpi (param.)	7, 1762
convertformat (param.)	7, 1762
distort (param.)	6, 718
draft (param.)	6, 718
exclude (param.)	7, 1762
extract (param.)	7, 1762
extractangle (param.)	7, 1762
extractapptex (param.)	7, 1762
extractdistort (param.)	7, 1762
extractformat (param.)	7, 1762
extractheight (param.)	7, 1762
extractpreamble (param.)	7, 1762
extractpretex (param.)	7, 1762
extractruns (param.)	7, 1762
extractscale (param.)	7, 1762
extractwidth (param.)	7, 1762

gsdevice (param.)	7, 1762	ext	216
gsopt (param.)	7, 1762	extension	216
height (param.)	6, 718	extract	8, 1142
inkscape (param.)	6, 718	extractapptex	8, 1683
inkscapearea (param.)	6, 718	extractdistort	8, 1587
inkscaledpi (param.)	6, 718	extractformat	8, 1174
inkscapeformat (param.)	6, 718	extractheight	8, 1587
inksapelatex (param.)	6, 718	extractkeepaspectratio	1587
inkscapeopt (param.)	6, 718	extractname	8, 1518
lastpage (param.)	6, 747	extractpath	8, 1518
latexopt (param.)	7, 1762	extractpostex	1683
magickoperator (param.)	7, 1762	extractpreamble	8, 1215
magicksetting (param.)	7, 1762	extractpreambleend	8, 1215
origin (param.)	6, 750	extractpretex	8, 1683
pretex (param.)	6, 718	extractruns	9, 1242
scale (param.)	6, 718	extractscale	8, 1587
svgextension (param.)	6, 718	extractwidth	8, 1587
width (param.)	6, 718	gsdevice	11, 1496
inkscape (opt.)	4, 56	gsexex	11, 1496
inkscapearea (opt.)	5, 163	gsopt	11, 1496
inkscapedensity (opt.)	177	height	5, 254
inkscapedpi (opt.)	5, 177	inkscape	4, 56
inkscapeexe (opt.)	5, 194	inkscapearea	5, 163
inkscapeformat (opt.)	5, 138	inkscapedensity	177
inkscapelatex (opt.)	5, 159	inkscapedpi	5, 177
inkscapename (opt.)	5, 223	inkscapeexe	5, 194
inkscapeopt (opt.)	5, 194	inkscapeformat	5, 138
inkscapepath (opt.)	4, 223	inkscapelatex	5, 159
K			
keepaspectratio (opt.)	254	inkscapename	5, 223
L			
lastpage (opt.)	6, 334	inkscapeopt	5, 194
latex (opt.)	159	inkscapepath	4, 223
latexexe (opt.)	9, 1259	keepaspectratio	254
latexext (opt.)	9, 1259	lastpage	6, 334
latexopt (opt.)	9, 1259	latex	159
M			
magickexe (opt.)	11, 1474	latexexe	9, 1259
magickoperator (opt.)	11, 1474	latexext	9, 1259
magicksetting (opt.)	11, 1474	latexopt	9, 1259
N			
name (opt.)	1518	magickexe	11, 1474
nottransparent (opt.)	3, 32	magickoperator	11, 1474
noxcolor (opt.)	3, 32	magicksetting	11, 1474
O			
off (opt.)	7, 136 , 1170	name	1518
on (opt.)	7, 136 , 1170	nottransparent	3, 32
options:		noxcolor	3, 32
apptex	5, 311	off	7, 136 , 1170
clean	9, 1719	on	7, 136 , 1170
clear	1719	path	1518
convert	10, 1346	pdf	1174
convertdensity	1449	pdflatex	1259
convertdpi	10, 1449	pdftepsopt	9, 1297
convertformat	10, 1422	pdftops	1297
distort	5, 254	pdftopsopt	9, 1297
draft	6, 348	png	1422
dvipsopt	9, 1297	postex	311
end	1215	preamble	1215
eps	1174	pretex	5, 311
exclude	9, 1740	pstoeptopt	9, 1297
		pstopdfopt	9, 1297
		scale	5, 254
		svgextension	5, 216
		svgpath	204
		tex	159
		usetransparent	3, 32
		usexcolor	3, 32
		width	5, 254

P

parameters:

angle- \includeinkscape	6, 807
angle- \includesvg	6, 750
apptex- \includeinkscape	6, 807
apptex- \includesvg	6, 718
clean- \includeinkscape	7, 1762
clean- \includesvg	7, 1762
convert- \includeinkscape	7, 1762
convert- \includesvg	7, 1762
convertdpi- \includeinkscape	7, 1762
convertdpi- \includesvg	7, 1762
convertformat- \includeinkscape	7, 1762
convertformat- \includesvg	7, 1762
distort- \includeinkscape	6, 807
distort- \includesvg	6, 718
draft- \includeinkscape	6, 807
draft- \includesvg	6, 718
exclude- \includeinkscape	7, 1762
exclude- \includesvg	7, 1762
extract- \includeinkscape	7, 1762
extract- \includesvg	7, 1762
extractangle- \includeinkscape	7, 1762
extractangle- \includesvg	7, 1762
extractapptex- \includeinkscape	7, 1762
extractapptex- \includesvg	7, 1762
extractdistort- \includeinkscape	7, 1762
extractdistort- \includesvg	7, 1762
extractformat- \includeinkscape	7, 1762
extractformat- \includesvg	7, 1762
extractheight- \includeinkscape	7, 1762
extractheight- \includesvg	7, 1762
extractpreamble- \includeinkscape	7, 1762
extractpreamble- \includesvg	7, 1762
extractpretex- \includeinkscape	7, 1762
extractpretex- \includesvg	7, 1762
extractruns- \includeinkscape	7, 1762
extractruns- \includesvg	7, 1762
extractscale- \includeinkscape	7, 1762
extractscale- \includesvg	7, 1762
extractwidth- \includeinkscape	7, 1762
extractwidth- \includesvg	7, 1762
gsdevice- \includeinkscape	7, 1762
gsdevice- \includesvg	7, 1762
gsopt- \includeinkscape	7, 1762
gsopt- \includesvg	7, 1762
height- \includeinkscape	6, 807
height- \includesvg	6, 718
inkscape- \includesvg	6, 718
inkscapearea- \includesvg	6, 718
inkscapedpi- \includesvg	6, 718
inkscapeformat- \includeinkscape	6, 807
inkscapeformat- \includesvg	6, 718
inksapelatex- \includeinkscape	6, 807
inksapelatex- \includesvg	6, 718
inkscapeopt- \includesvg	6, 718
lastpage- \includeinkscape	6, 807
lastpage- \includesvg	6, 747
latexopt- \includeinkscape	7, 1762
latexopt- \includesvg	7, 1762
magickoperator- \includeinkscape	7, 1762
magickoperator- \includesvg	7, 1762
magicksetting- \includeinkscape	7, 1762

magicksetting- \includesvg	7, 1762
origin- \includeinkscape	6, 807
origin- \includesvg	6, 750
pretex- \includeinkscape	6, 807
pretex- \includesvg	6, 718
scale- \includeinkscape	6, 807
scale- \includesvg	6, 718
svgextension- \includesvg	6, 718
width- \includeinkscape	6, 807
width- \includesvg	6, 718
path (opt.)	1518
pdf (opt.)	1174
pdflatex (opt.)	1259
pdftoept (opt.)	9, 1297
pdftops (opt.)	1297
pdftops (opt.)	9, 1297
png (opt.)	1422
postex (opt.)	311
preamble (opt.)	1215
pretex (opt.)	5, 311
pstoeps (opt.)	9, 1297
pstopdf (opt.)	9, 1297

S

scale (opt.)	5, 254
\setsvg	701
\svg@append@input@path	411
\svg@box	14
\svg@deactivate@dq	351
\svg@deprecated@key	24
\svg@deprecated@param	827
\svg@dummy@key	1115
\svg@extension@parse	566
\svg@extension@@parse	566
\svg@extract	1804
\svg@file@base	489
\svg@file@ext	216
\svg@file@missing	602
\svg@file@name	489
\svg@file@path	489
\svg@file@suffix	489
\svg@filename@parse	529
\svg@get@lastpage	932
\svg@get@path	489
\svg@iffilenewer	648
\svg@ifvalueisrelax	478
\svg@includegraphics@file	1097
\svg@includegraphics@patched	1097
\svg@includegraphics@saved	1065
\svg@ink@area	163
\svg@ink@cmd	924
\svg@ink@dpi	177
\svg@ink@exe	194
\svg@ink@format	138
\svg@ink@latex	159
\svg@ink@mode	56
\svg@ink@opt	194
\svg@ink@run	849
\svg@input	983
\svg@input@path	703
\svg@@input	983
\svg@local@param@def	680
\svg@local@param@set	680
\svg@local@param@use	680
\svg@normalize@path	452
\svg@normalize@@path	452

<code>\svg@out@base</code>	223	<code>\svgx@ifkeyandval</code>	2294
<code>\svg@out@name</code>	223	<code>\svgx@latex@exe</code>	1259
<code>\svg@out@path</code>	223	<code>\svgx@latex@ext</code>	1259
<code>\svg@param@apptex</code>	311	<code>\svgx@latex@opt</code>	1259
<code>svg@param@lastpage</code> (counter)	334	<code>\svgx@magick@cmd</code>	2215
<code>\svg@param@pretex</code>	311	<code>\svgx@magick@exe</code>	1474
<code>\svg@param@scale</code>	254	<code>\svgx@magick@opr</code>	1474
<code>\svg@param@width</code>	254	<code>\svgx@magick@set</code>	1474
<code>\svg@patches</code>	1065	<code>\svgx@move</code>	2244
<code>\svg@pictur@patched</code>	1074	<code>\svgx@onlywindows</code>	2292
<code>\svg@picture@savd</code>	1065	<code>svgx@out@count</code> (counter)	1518
<code>\svg@quotes@check</code>	387	<code>\svgx@out@name</code>	1518
<code>\svg@quotes@@check</code>	387	<code>\svgx@out@path</code>	1518
<code>\svg@quotes@remove</code>	362	<code>\svgx@out@sec</code>	2131
<code>\svg@quotes@@remove</code>	362	<code>\svgx@param@apptex</code>	1683
<code>\svg@remove@leadingchar</code>	395	<code>\svgx@param@distort</code>	1587
<code>\svg@sanitize@dq</code>	356	<code>\svgx@param@pretex</code>	1683
<code>\svg@set@input@path</code>	411	<code>\svgx@param@scale</code>	1587
<code>\svg@shell@mkdir</code>	2463	<code>\svgx@param@width</code>	1587
<code>\svg@shell@mmdir</code>	2463	<code>\svgx@pdftoeps@exe</code>	1297
<code>\svg@shell@mv</code>	2463	<code>\svgx@pdftoeps@opt</code>	1297
<code>\svg@shell@mv</code>	2463	<code>\svgx@pdftops@exe</code>	1297
<code>\svg@shell@rm</code>	2463	<code>\svgx@pdftops@opt</code>	1297
<code>\svg@shell@rm</code>	2463	<code>\svgx@preamble</code>	1215
<code>\svg@tempa</code>	14	<code>\svgx@pstoeps@exe</code>	1297
<code>\svg@tempb</code>	14	<code>\svgx@pstoeps@opt</code>	1297
<code>\svg@wrn@scale</code>	966	<code>\svgx@pstopdf@exe</code>	1297
<code>svgextension</code> (opt.)	5 , 216	<code>\svgx@pstopdf@opt</code>	1297
<code>\svghidepreambleend</code>	9 , 1784	<code>\svgx@read@line</code>	1804
<code>\svghidepreamblestart</code>	9 , 1784	<code>\svgx@read@preamble@from</code>	2159
<code>\svgpath</code>	4 , 703	<code>\svgx@read@preamble@skip</code>	2159
<code>svgpath</code> (opt.)	204	<code>\svgx@read@preamble@till</code>	2159
<code>\svgsetup</code>	3 , 7 , 701	<code>svgx@runs</code> (counter)	1242
<code>\svgx@clean</code>	1719	<code>\svgx@setbox</code>	2377
<code>\svgx@cnv@cmd</code>	1346	<code>\svgx@setformatkey</code>	2344
<code>\svgx@cnv@dpi</code>	1449	<code>\svgx@stream@in</code>	1804
<code>\svgx@cnv@format</code>	1422	<code>\svgx@stream@out</code>	1804
<code>\svgx@cnv@get@dpi</code>	2298	<code>\svgx@useformatkey</code>	2344
<code>\svgx@cnv@get@informat</code>	2196	<code>\svgxoutputbox</code>	2423
<code>\svgx@cnv@informat</code>	2196	<code>\svgxsetbox</code>	2377
<code>\svgx@documentclass</code>	2154	<code>\svgxsetpapersize</code>	2392
<code>\svgx@dvi@exe</code>	1297		
<code>\svgx@dvi@opt</code>	1297		
<code>\svgx@endpreamble</code>	1215		
<code>\svgx@format</code>	1174		
<code>\svgx@get@out@sec</code>	2131		
<code>\svgx@gs@cmd</code>	2215		
<code>\svgx@gs@device</code>	1496		
<code>\svgx@gs@exe</code>	1496		
<code>\svgx@gs@opt</code>	1496		
<code>\svgx@ifinlist</code>	2280		

	T		
<code>tex</code> (opt.)		159	
	U		
<code>usetransparent</code> (opt.)		3 , 32	
<code>usexcolor</code> (opt.)		3 , 32	
	W		
<code>width</code> (opt.)		5 , 254	

Change History

v1.0

General

initial version by Philip Ilten [2](#)

v2.00

General

new maintainer: Falk Hanisch [2](#)

package **subfig** not required anymore [2](#)

re-implementation from scratch [2](#)

support of subfigures stopped due to the

huge number of packages which deal with this topic and the large variety of implementing this functionality; naming exported graphics after their consecutive numbering can't be ensured for all variants of subfigures, so it's neglected [2](#)

Implementation

`clean` (opt.): changes, file list possible [1719](#)

`convert` (opt.): changed/extended [1346](#)

convertdpi (opt.): new	1449
convertformat (opt.): new	1422
draft (opt.): new	348
dvipsopt (opt.): new	1297
end (opt.): deprecated	1215
eps (opt.): deprecated	1174
extract (opt.): new	1142
extractapptex (opt.): new	1683
extractformat (opt.): new	1174
extractheight (opt.): new	1587
extractname (opt.): new	1518
extractpath (opt.): new	1518
extractpreamble (opt.): new	1215
extractpreambleend (opt.): new	1215
extractpretex (opt.): new	1683
extractruns (opt.): new	1242
extractscale (opt.): new	1587
extractwidth (opt.): new	1587
gsdevice (opt.): new	1496
gsexec (opt.): new	1496
gsopt (opt.): new	1496
height (opt.): new	254
\includeinkscape: new	766
\includesvg:	
changes, especially to optional	
parameters	715
angle (param.): new	750
draft (param.): new	718
height (param.): new	718
inkscape (param.): new	718
inkscapearea (param.): new	718
inkscapedpi (param.): new	718
inkscapeformat (param.): new	718
inkscapelatex (param.): new	718
inkscapeopt (param.): new	718
lastpage (param.): new	747
origin (param.): new	750
scale (param.): new	718
inkscape (opt.): changed/extended	56
inkscapearea (opt.): new	163
inkscapedpi (opt.): new	177
inkscapeexe (opt.): new	194
inkscapeformat (opt.): new	138
inkscapelatex (opt.): new	159
inkscapename (opt.): new	223
inkscapeopt (opt.): new	194
inkscapepath (opt.): new	223
lastpage (opt.): new	334
latexexe (opt.): new	1259
latexext (opt.): new	1259
latexopt (opt.): new	1259
magickexe (opt.): new	1474
magickoperator (opt.): new	1474
magicksetting (opt.): new	1474
name (opt.):	
deprecated	1518
support of subfig removed	1518
notransparent (opt.): new	32
noxcolor (opt.): new	32
off (opt.): new	136, 1170
on (opt.): new	136, 1170
path (opt.): deprecated	1518
pdf (opt.): deprecated	1174
pdflatex (opt.): deprecated	1259
pdfstoeps (opt.): new	1297
pdftops (opt.): deprecated	1297
pdftopsopt (opt.): new	1297
png (opt.): deprecated	1422
postex (opt.): deprecated	311
preamble (opt.): deprecated	1215
pstoeps (opt.): new	1297
pstopdfopt (opt.): new	1297
scale (opt.): new	254
\setsvg: deprecated	701
\svghidepreambleend: new	1784
\svghidepreamblestart: new	1784
\svgpath: new	703
svgpath (opt.): deprecated	204
\svgsetup: new	701
usetransparent (opt.): new	32
usexcolor (opt.): new	32
v2.00a	
Implementation	
\svgxsetpapersize: Bug fix for	
checking stock- and mediasizes	2392
v2.00b	
Implementation	
latex (opt.): new, alternative key for	
inkscapelatex	159
tex (opt.): new, alternative key for	
inkscapelatex	159
v2.01	
General	
new option svgextension to change the	
format of files exported by Inkscape	
from svg to a custom one	2
usage of <code>\input{<tex filename>}</code> within	
Inkscape graphics locates files in all	
declared searched folders	2
Implementation	
\includesvg:	
svgextension (param.): new	718
inkscape (opt.): using <code>\trim@spaces</code>	56
\svg@append@input@path: new	411
\svg@get@path:	
using <code>\svg@set@input@path</code>	489
using <code>\trim@spaces</code>	489
\svg@set@input@path: new	411
svgextension (opt.): new due to user	
request	216
v2.02	
General	
distortion of included and extracted	
graphics supported with options	
distort (or keepaspectratio) and	
extractdistort as well as rotation	
for extractions (extractangle)	2
fixed errors with active double quotes	
from babel in path arguments	2
multiple dots within file names possible	2
package trimspaces required	2
Implementation	
distort (opt.): new	254
extractdistort (opt.): new	1587
extractname (opt.): usage of	
\svg@quotes@remove	1518
extractpath (opt.): usage of	
\svg@sanitize@dq	1518
\if@svgx@standalone: new	2377

<code>\includeinkscape:</code>		usage of <code>\svg@remove@leadingchar</code> 216
usage of <code>\svg@extension@parse</code> ..	766	<code>\svgpath:</code> parse argument for enclosing braces and provide if necessary .. 703
<code>extractddistort</code> (param.): new ..	1762	<code>\svgx@setbox:</code> new .. 2377
<code>\includesvg:</code>		<code>\svgxsetbox:</code> late execution of <code>\svgxsetpapersize</code> .. 2377
switched to <code>\svg@filename@parse</code>	715	
<code>angle</code> (param.): validation of argument ..	750	
<code>distort</code> (param.): new ..	718	
<code>extractangle</code> (param.): new ..	1762	
<code>extractddistort</code> (param.): new ..	1762	
<code>inkscape</code> (opt.): usage of <code>\svg@sanitize@dq</code> ..	56	
<code>inkscapepath</code> (opt.): usage of <code>\svg@sanitize@dq</code> ..	223	
<code>keepaspectratio</code> (opt.): new ..	254	
<code>\svg@append@input@path:</code> avoid duplicates in <code>\input@path</code> ..	411	
<code>\svg@deactivate@dq:</code> new ..	351	
<code>\svg@extension@parse:</code> new ..	566	
<code>\svg@extension@@@parse:</code> new ..	566	
<code>\svg@file@missing:</code> notify svg file when missing exported files ..	602	
<code>\svg@filename@parse:</code> usage of <code>\svg@extension@parse</code> ..	529	
usage of <code>\svg@remove@leadingchar</code>	529	
usage of <code>\svg@sanitize@dq</code> ..	529	
<code>\svg@local@param@set:</code> reasonable value for key <code>distort</code> ..	680	
<code>\svg@normalize@path:</code> usage of <code>\svg@deactivate@dq</code> ..	452	
<code>\svg@quotes@remove:</code> calling <code>\svg@quotes@check</code> ..	362	
usage of <code>\svg@sanitize@dq</code> ..	362	
<code>\svg@remove@leadingchar:</code> new ..	395	
<code>\svg@sanitize@dq:</code> new ..	356	
<code>\svg@set@input@path:</code> usage of <code>\svg@deactivate@dq</code> ..	411	
<code>svgextension</code> (opt.): usage of <code>\svg@quotes@remove</code> ..	216	
		v2.02a
		General fix bug for package polyglossia which fakes babel poorly .. 2
		Implementation <code>\svg@deactivate@dq:</code> bug fix for polyglossia .. 351
		v2.02b
		General fix bug for package tikzscale which changes <code>\includegraphics</code> globally .. 2
		Implementation <code>\svg@patches:</code> fix bug for package tikzscale: store original definitions of <code>\picture</code> and <code>\includegraphics</code> right after loading package svg .. 1065
		v2.02c
		General fix bugs with current kernel (2019/10/01) regarding file name parsing .. 2
		v2.02d
		General fix bugs with current kernel (2019/10/01) regarding file name parsing, see https://github.com/ mrpiggi/svg/issues/16 .. 2
		Implementation <code>\svg@ifilenewer:</code> use <code>\filemoddate</code> with Xe _{La} TeX, see https://github. com/mrpiggi/svg/issues/12 .. 648