

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

Philip Ilten (2012–2016)

Falk Hanisch (2017–)

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

hanisch.latex@outlook.com

v2.00 (2017/03/27)

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 svg	3
2.1.	General settings	4
2.2.	Options for the invocation of <i>Inkscape</i>	4
2.3.	Options for the graphic inclusion	6
2.4.	Including SVG files	6
3.	Usage of package svg-extract	7
3.1.	General settings	8
3.2.	Extract independent graphic files	8
3.3.	Convert extracted graphic files	10
3.3.1.	Settings for the invocation of <i>ImageMagick</i>	11
3.3.2.	Settings for the invocation of <i>Ghostscript</i>	12
4.	Example	12
5.	Troubleshooting and reporting issues	14
6.	Include SVG files created with <i>ROOT</i>	14

II.	Implementation	17
A.	Initialization	17
B.	Including SVG files with package <i>svg</i>	18
B.1.	Options	18
B.1.1.	The invocation of <i>Inkscape</i>	18
B.1.2.	Setting input folder	22
B.1.3.	Setting output folder	22
B.1.4.	Options for the inclusion of graphics	23
B.2.	Handling path information	24
B.3.	Optional Parameters for user commands	30
B.4.	User commands	30
B.5.	Auxiliary macros	33
B.6.	Patches	38
C.	Extracting independent graphic files with package <i>svg-extract</i>	40
C.1.	Options	40
C.1.1.	Controlling the extract process	40
C.1.2.	Invoking external application for graphic conversion	44
C.1.3.	Setting output folder	48
C.1.4.	Options for the extraction of graphics	49
C.1.5.	Miscellaneous options	51
C.2.	User commands	52
C.3.	Auxiliary macros	53
C.4.	Commands for the separate auxiliary <i>L^AT_EX</i> -file	66
D.	Processing Options	67
E.	Macros for file access	68
	Index	69
	Change History	73

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 *L^AT_EX*, and an image component that can be imported as a PDF file. For further information see the documentation of *svg-inkscape* on CTAN¹. The procedure described therein is taken up and consistently expanded. Thus, it is now possible to include a SVG file into a *L^AT_EX* document where the text within the SVG graphic will be rendered natively by *L^AT_EX*.

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

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**.

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 packages are:

scrbase for the definition and handling of options in key-value-syntax
ifluatex, **ifpdf**, **ifxetex** for flow control depending on the used L^AT_EX engine
pdftexcmds, **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
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
`usetransparent` (opt.) listed options are intended to prevent these packages from loading. They are the only
`noxcolor` (opt.) options which have to be given while loading the **svg** package. All supported boolean values
`notransparent` (opt.) (true/on/yes/false/off/no) can be assigned to `usexcolor` and `usetransparent`, while `noxcolor` and `notransparent` don't accept any value.

```
\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 set to `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. Unfortunately a primitive like `\pdffilemoddate` is missing for XeTeX, so with this engine, the behaviour will be the same as `inkscape=forced`.

`forced/force/overwrite`

The ***Inkscape*** export will definitely be done, any already existing exported file will 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 `inkscaelatex=true/false`

`drawing/page`

see `inkscapearea=drawing/page`

`⟨integer⟩dpi`

see `inkscapecdpi=⟨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 L^AT_EX 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.
- inkscapeexe** (opt.) For the inclusion of 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 L^AT_EX 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 L^AT_EX 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 L^AT_EX.
- 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.
- inkscapedpi** (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 **inkscapedpi=\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.

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

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 by the `height` option. If both the width and height are specified, the figure won't be distort but scaled such that neither of the specified dimensions is exceeded. If `width` and/or `height` `scale` (opt.) once have been set, this can be reversed by setting them to `0pt` 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 `apptex` (opt.) color commands. The options `pretex` and `apptex` are provided where the L^AT_EX code given to `pretex` is included before the graphic file and `apptex` right afterwards. For example, to change the size of the included text one could use:

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

`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](#)³ concerning the L^AT_EX export has been reported for **Inkscape** 0.91. It may happen that within the L^AT_EX file exported by **Inkscape**, it is 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 `\includsvg` or `\includeinkscape`. For details, see the description of the respective commands.

2.4. Including SVG files

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

```
\includsvg[<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, `inkscapeformat` (param.) where any given file extension will be replaced with `.svg` ruthlessly. If this file is not located `inkscaelatex` (param.) in the current working directory but elsewhere on your file system, the command `\svgpath` `inkscapearea` (param.) could be used to specify this path. It is recommended to avoid any spaces and/or quotes `inkscapedpi` (param.) respectively `\dq` both in paths and file names. Especially when DVI output is active using `inkscapeopt` (param.) quotes will certainly cause an error.

`width` (param.) The command `\includsvg` is intended to do an automated export with **Inkscape** at first, `height` (param.) where the given SVG file is exported to a PDF/EPS/PS/PNG file (see `inkscapeformat`) and `scale` (param.) perhaps a correlating L^AT_EX file (see `inkscaelatex`). The export with **Inkscape** is only `pretex` (param.) invoked, if the SVG file is newer than the exported graphic file or latter doesn't exist at all.⁴ `apptex` (param.) Once the export has been done, the graphic file and maybe the L^AT_EX file are included. `draft` (param.)

All previously described options can also be used as optional parameters to `\includsvg` and do have the same effect as described before. However, the optional parameters specified have an effect only once when `\includsvg` is executed and remain unchanged afterwards.

³<https://bugs.launchpad.net/ubuntu/+source/inkscape/+bug/1417470>

⁴Due to the lack of XeTeX to compare file modification dates, using this L^AT_EX engine leads to **Inkscape** exports with every run unless `inkscape=false` is used.

<code>lastpage</code> (param.)	In addition to the use of boolean values, the parameter <code>lastpage</code> 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 <code>inkscapeformat=pdf</code> is set.
<code>angle</code> (param.) <code>origin</code> (param.)	Both parameters correlate to the identically named parameters of the <code>\includegraphics</code> command provided by the graphicx package. However, unlike to <code>\includegraphics</code> , parameters <code>angle</code> and <code>origin</code> are <i>always evaluated after</i> the parameters <code>width</code> , <code>height</code> and <code>scale</code> by <code>\includesvg</code> , regardless of the used order of the given parameters. This is mainly due to the inclusion of the L ^A T _E X files corresponding to the graphic files generated by Inkscape .
<code>\includeinkscape</code>	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 <code>\includeinkscape</code> instead of <code>\includesvg</code> . <div style="border: 1px solid black; padding: 5px; margin-top: 5px;"> <code>\includeinkscape[⟨parameters⟩]{⟨graphic filename⟩}</code> </div>
<code>inkscapeformat</code> (param.) <code>inkscapelatex</code> (param.) <code>width</code> (param.) <code>height</code> (param.) <code>scale</code> (param.) <code>pretex</code> (param.) <code>apptex</code> (param.) <code>draft</code> (param.) <code>lastpage</code> (param.) <code>angle</code> (param.) <code>origin</code> (param.)	You can use it similar to <code>\includesvg</code> but <code>{⟨graphic filename⟩}</code> has to be the filename of the already exported graphic file. If a valid file extension (<code>.pdf/.eps/.ps/.png</code>) is given, the current setting for <code>inkscapeformat</code> is overwritten. It's even possible to specify a file extension like <code>.pdf_tex</code> to activate <code>inkscapelatex</code> . Furthermore, all optional parameters for <code>\includeinkscape</code> do have the same effect as described before for command <code>\includesvg</code> once when <code>\includeinkscape</code> is executed and remain unchanged afterwards.

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 format(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 off 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` 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 commands are executed and remain unchanged afterwards.

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, a given file extension is ignored. 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
`extractheight` (opt.) within the document. For example, a SVG file should cover the entire text width within the
`extractscale` (opt.) document but be extracted to a fixed width, this can be done with:

```
\includesvg[width=\textwidth,extractwidth=500pt]{<svg filename>}
```

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.

`extractpreamble` (opt.) Within the included and extracted SVG files any \LaTeX macro can be used either defined by
`extractpreambleend` (opt.) 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 known within this file. Consequently, the preamble of the current \LaTeX 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 \LaTeX document is used, there are maybe packages included
`\svghidepreambleend` or some parts within the preamble, which should not be used within the separate auxiliary \LaTeX file. These parts can be excluded if they are enclosed by `\svghidepreamblestart` and `\svghidepreambleend`.

For example, your current \LaTeX 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 \LaTeX 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 \LaTeX file, it's maybe necessary to do multiple \LaTeX 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, it is set to either **pdf_latex**, **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 provided by most of the **L^AT_EX 2_ε** distributions are maybe invoked. These are **dvips**, **ps2eps**, **ps2pdf** and/or **pdftops** and can't be changed. It's only possible to specify additional switches for every single tool with **dvipsopt**, **pstoepsopt**, **pstopdfopt**, **pdftoepsopt** and **pdftopsopt**.
- 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. Therefor 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 options 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 unspecific 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](http://www.imagemagick.org/script/command-line-processing.php)⁵.

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

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
`gsdevice` (opt.) systems and `gswin64c` or `gswin32c` on Windows. The executable can be changed with option
`gsopt` (opt.) `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{selinput}\SelectInputMappings{adieresis={ä},germandbls={ß}}  
\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,pretext=\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}  
\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.

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 \LaTeX by Johan B. C. Engelen available as package `svg-inkscape` on CTAN.

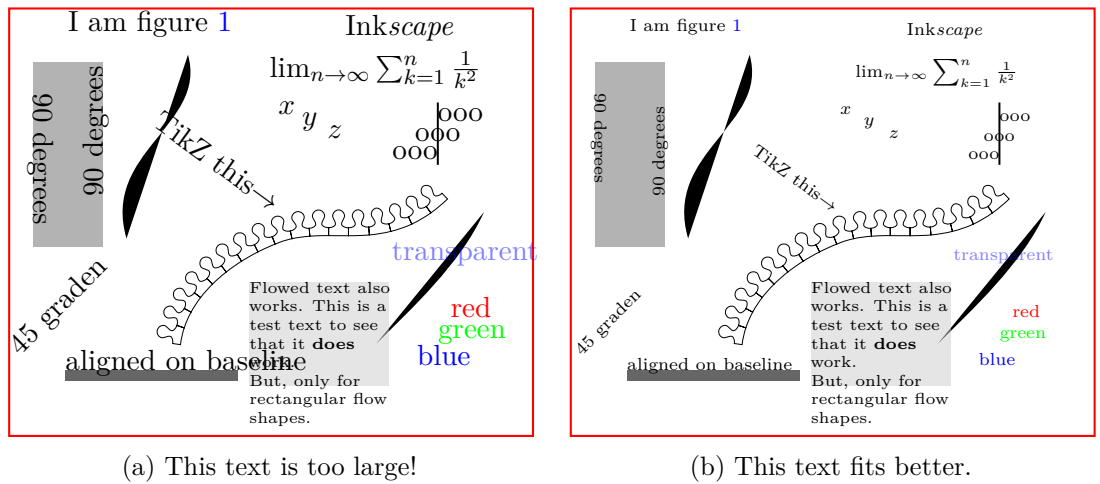


Figure 1: An example figure with L^AT_EX support

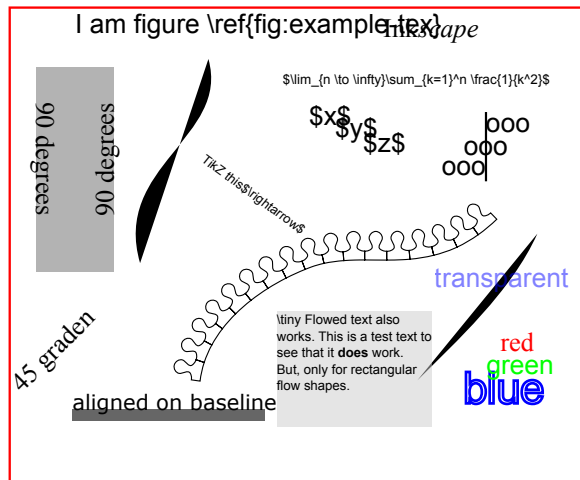


Figure 2: The same example figure without L^AT_EX support

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 L^AT_EX 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 **resize** 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 pdfT_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](http://tex.stackexchange.com/questions/76273/)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

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

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

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

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

```
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("LEG02");

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

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

}
```

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

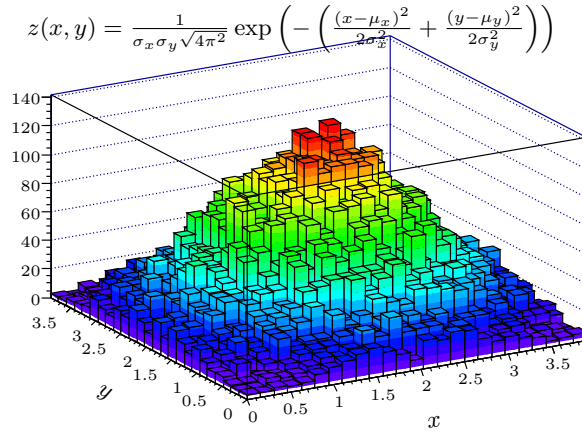


Figure 3: Rendering of a **ROOT** plot—no more *Comic CERNs*

The plot was then included within this document using the following L^AT_EX 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}}%
\label{fig:root}%
\end{figure}
```

which includes the graphic as well as the L^AT_EX 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 L^AT_EX!

Part II.

Implementation

A. Initialization

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{ifxetex}[2010/09/12]
4 \RequirePackage{ifluatex}[2016/05/16]
5 \RequirePackage{ifpdf}[2016/05/14]
6 \RequirePackage{pdftexcmds}[2016/05/21]
7 \RequirePackage{shellesc}[2016/06/07]
8 \RequirePackage{graphicx}[1999/02/16]
9 \*base)
10 \*extract)
11 \RequirePackage{svg}[2017/03/27]
12 \*extract)

```

With the interface provided by package **scrbase** 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.

```

13 \DefineFamily{SVG}
14 \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.

```

15 \*base)
16 \newcommand*\svg@deprecated@key[3][svg]{%
17   \PackageWarning{#1}{%
18     The option key ‘#2’ is deprecated.\MessageBreak%
19     It’s recommended to use ‘#3’\MessageBreak%
20     instead%
21   }}%
22 \FamilyOptions{SVG}{#3}%
23 }
24 \*base)

```

<code>\svg@tempa</code> <code>\svg@tempb</code> <code>\if@svg@tempswa</code> <code>\svg@tempcnt</code>	Internal temporary macros. <pre> 25 *base) 26 \newcommand*\svg@tempa{} 27 \newcommand*\svg@tempb{} 28 \newif\if@svg@tempswa </pre>
---	--

```

29 \newcount\svg@tempcnt
30 \</base>

```

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

B.1. Options

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 (opt.)</code> <code>noxcolor (opt.)</code> <code>\if@svg@use@xcolor</code> <code>usetransparent (opt.)</code> <code>nottransparent (opt.)</code> <code>\if@svg@use@transparent</code>	Options for preventing packages xcolor and transparent to be loaded. <pre> 31 \newif\if@svg@use@xcolor 32 \FamilyBoolKey{SVG}{usexcolor}{@svg@use@xcolor} 33 \DeclareOption{noxcolor}{\FamilyOptions{SVG}{usexcolor=false}} 34 \newif\if@svg@use@transparent 35 \FamilyBoolKey{SVG}{usetransparent}{@svg@use@transparent} 36 \DeclareOption{nottransparent}{\FamilyOptions{SVG}{usetransparent=false}} </pre>
--	--

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

```

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

```

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 (opt.)</code> <code>\svg@ink@mode</code>	The intension of option inkscape is to control the running behaviour of Inkscape . It can be switched off at all or invoked only if necessary or the command line call can be forced with every L ^A T _E X run. Additionally, option inkscape can be used as wrapper for options inkscapeformat , inkscapelatex , inkscapearea and inkscapedpi , which are declared later. <pre> 55 \newcommand*\svg@ink@mode{} 56 \DefineFamilyKey{SVG}{inkscape}[true]{% 57 \lowercase{\def\svg@tempa{#1}}% </pre>
--	---

```

58 \FamilySetNumerical{SVG}{inkscape}{svg@tempa}{%
59   {false}{0},{off}{0},{no}{0},{%
60   {true}{1},{on}{1},{yes}{1},{onlynewer}{1},{newer}{1},{%
61   {force}{2},{forced}{2},{overwrite}{2},{%
62   {pdf}{3},{eps}{4},{ps}{5},{png}{6},{%
63   {drawing}{7},{crop}{7},{%
64   {page}{8},{nocrop}{8},{%
65   {tex}{9},{latex}{9},{exportlatex}{9},{latexexport}{9},{%
66   {notex}{10},{nolatem}{10},{noexportlatex}{10},{nolatemexport}{10},{%
67   {latexnoexport}{10},{raw}{10},{plain}{10},{simple}{10}%
68 }{\svg@tempa}%
69 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
70   \ifnum\svg@tempa<\thr@@\relax%
71     \let\svg@ink@mode\svg@tempa%
72   \else%
73     \ifcase\svg@tempa\relax\or\or\or% pdf
74       \FamilyOptions{SVG}{inkscapeformat=pdf}%
75     \or% eps
76       \FamilyOptions{SVG}{inkscapeformat=eps}%
77     \or% ps
78       \FamilyOptions{SVG}{inkscapeformat=ps}%
79     \or% png
80       \FamilyOptions{SVG}{inkscapeformat=png}%
81     \or% drawing
82       \FamilyOptions{SVG}{inkscapearea=drawing}%
83     \or% page
84       \FamilyOptions{SVG}{inkscapearea=page}%
85     \or% tex
86       \FamilyOptions{SVG}{inkscapecatex=true}%
87     \or% notex
88       \FamilyOptions{SVG}{inkscapecatex=false}%
89     \fi%
90   \fi%

```

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

```

91 \else% dpi
92   \def\svg@tempa##1dpi##2\@nil{%
93     \ifstr{##2}{dpi}{\FamilyOptions{SVG}{inksapedpi=##1}}{ }%
94   }%
95   \lowercase{\svg@tempa#1dpi\@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.

```

96 \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`.

```

97   \def\svg@tempa##1-##2\@nil{%
98     \IfArgIsEmpty{##2}{\def\svg@tempb{}}{ }%
99     \def\svg@tempa##1####1\@nil{\def\svg@tempb{####1}}%
100    \svg@tempa#1\@nil%
101  }%
102  \def\svg@tempa{##1}%
103 }%

```

```

104 \svg@tempa#1-\@nil%
105 \PackageWarning{svg}{%
106   Setting the executable%
107   \ifx\svg@tempb\@empty\else%
108     \space and associated options%
109   \fi%
110   \MessageBreak%
111   for Inkscape should be done with options\MessageBreak%
112   'inkscapeexe=\svg@tempa'%
113   \ifx\svg@tempb\@empty\else%
114     \MessageBreak and 'inkscapeopt=\svg@tempb'%
115   \fi.\MessageBreak%
116   Nevertheless, this was done by now anyway%
117 }%
118 \edef\svg@tempa{%
119   \noexpand\FamilyOptions{SVG}{inkscapeexe=\svg@tempa}%
120   \ifx\svg@tempb\@empty\else%
121     \noexpand\FamilyOptions{SVG}{inkscapeopt=\svg@tempb}%
122   \fi%
123 }%
124 \svg@tempa%
125 \fi%
126 \fi%
127 }

```

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

```

128 \DeclareOption{on}{\FamilyOptions{SVG}{inkscape=true}}
129 \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.

```

130 \newcommand*\svg@ink@format{pdf}
131 \ifxetex\else\ifpdf\else
132   \renewcommand*\svg@ink@format{eps}
133 \fi\fi
134 \DefineFamilyKey{SVG}{inkscapeformat}{%
135   \lowercase{\def\svg@tempa{#1}}%
136   \FamilySetNumerical{SVG}{inkscapeformat}{\svg@tempa}{%
137     {pdf}{0},{eps}{1},{ps}{2},{png}{3}}%
138   }\svg@tempa}%
139 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
140   \ifcase\svg@tempa\relax% latex
141     \renewcommand*\svg@ink@format{pdf}%
142   \or% eps
143     \renewcommand*\svg@ink@format{eps}%
144   \or% ps
145     \renewcommand*\svg@ink@format{ps}%
146   \or% png
147     \renewcommand*\svg@ink@format{png}%
148   \fi%
149 \fi%
150 }

```


`inkscapectex` (opt.) This options controls whether the **Inkscape** export will be invoked with or without the generation of a separate L^AT_EX file.

```

151 \newif\if@svg@ink@latex
152 \FamilyBoolKey{SVG}{inkscapectex}{@svg@ink@latex}

```

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

```

153 \newcommand*\svg@ink@area{}
154 \DefineFamilyKey{SVG}{inkscapearea}{%
155   \FamilySetNumerical{SVG}{inkscapearea}{svg@tempa}{%
156     {drawing}{0},{crop}{0},{%
157     {page}{1},{nocrop}{1}%
158   }{#1}%
159   \ifx\FamilyKeyState\FamilyKeyStateProcessed%
160     \ifcase\svg@tempa\relax% drawing
161       \renewcommand*\svg@ink@area{-D}%
162     \else% page
163       \renewcommand*\svg@ink@area{-C}%
164     \fi%
165   \fi%
166 }

```

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

`inkscapectdensity` (opt.)

```

167 \newcommand*\svg@ink@dpi{}
168 \let\svg@ink@dpi\relax
169 \DefineFamilyKey{SVG}{inkscapectpi}{%
170   \FamilyKeyStateUnknownValue%
171   \svg@ifvalueisrelax{#1}{%
172     \let\svg@ink@dpi\relax%
173     \FamilyKeyStateProcessed%
174   }{%
175     \def\svg@tempa##1dpi##2\@nil{\def\svg@tempa{##1}}%
176     \lowercase{\svg@tempa#1dpi\@nil}%
177     \ifnumber{\svg@tempa}{%
178       \edef\svg@ink@dpi{\svg@tempa}%
179       \FamilyKeyStateProcessed%
180     }{}%
181   }%
182 }
183 \DefineFamilyKey{SVG}{inkscapectdensity}{\FamilyOptions{SVG}{inkscapectpi=#1}}

```

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

`inkscapeopt` (opt.)

```

184 \newcommand*\svg@ink@exe{inkscape}
185 \DefineFamilyKey{SVG}{inkscapeexe}{%
186   \renewcommand*\svg@ink@exe{#1}%
187   \FamilyKeyStateProcessed%
188 }
189 \newcommand*\svg@ink@opt{}
190 \DefineFamilyKey{SVG}{inkscapeopt}{%
191   \renewcommand*\svg@ink@opt{#1}%
192   \FamilyKeyStateProcessed%
193 }

```

B.1.2. Setting input folder

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

```
194 \DefineFamilyKey{SVG}{svgpath}{%
195   \PackageWarning{svg}{%
196     The key 'svgpath' is deprecated. It's recommended\MessageBreak%
197     to use '\string\svgpath' instead%
198   }%
199   \ifx\svgpath\@undefined%
200     \AtEndOfPackage{\svgpath{{#1}}}%
201   \else%
202     \svgpath{{#1}}%
203   \fi%
204   \FamilyKeyStateProcessed%
205 }
```

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 `inkscapename` the name of the exported file itself can be changed.

```
inkscapepath (opt.)
inkscapename (opt.)
\svg@out@path
\svg@out@name
\svg@out@base
206 \newcommand*\svg@out@path{}
207 \newcommand*\svg@out@name{\svg@file@name\svg@file@suffix}
208 \newcommand*\svg@out@base{\svg@out@path\svg@out@name.\svg@ink@format}
209 \DefineFamilyKey{SVG}{inkscapepath}{%
210   \FamilySetNumerical{SVG}{inkscapepath}{svg@tempa}{%
211     {svgpath}{0},{svgdir}{0},%
212     {svgsubpath}{1},{svgsubdir}{1},%
213     {basepath}{2},{basedir}{2},{jobpath}{2},{jobdir}{2},%
214     {basesubpath}{3},{basesubdir}{3},{jobsubpath}{3},{jobsubdir}{3}%
215   }{#1}%
216   \ifx\FamilyKeyState\FamilyKeyStateProcessed%
217     \ifcase\svg@tempa\relax% svgpath
218       \renewcommand*\svg@out@path{\svg@file@path}%
219     \or% svgsubpath
220       \renewcommand*\svg@out@path{\svg@file@path svg-inkscape/}%
221     \or% basepath
222       \renewcommand*\svg@out@path{./}%
223     \or% basesubpath
224       \renewcommand*\svg@out@path{./svg-inkscape/}%
225     \fi%
226   \else%
227     \renewcommand*\svg@out@path{#1}%
228     \svg@normalize@path{\svg@out@path}%
229     \FamilyKeyStateProcessed%
230   \fi%
231 }
232 \DefineFamilyKey{SVG}{inkscapename}{%
233   \renewcommand*\svg@out@name{#1\svg@file@suffix}%
234   \FamilyKeyStateProcessed%
235 }
```

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.

<code>width</code> (opt.)	These options determine the size of the included graphics. The usage of <code>\relax</code> as value resets the respective option to the default behavior.
<code>\svg@param@width</code>	236 <code>\newcommand*\svg@param@width{\z@}</code>
<code>height</code> (opt.)	237 <code>\DefineFamilyKey{SVG}{width}{%</code>
<code>\svg@param@width</code>	238 <code>\FamilyKeyStateUnknownValue%</code>
<code>scale</code> (opt.)	239 <code>\svg@ifvalueisrelax{#1}{%</code>
<code>\svg@param@scale</code>	240 <code>\renewcommand*\svg@param@width{\z@}%</code>
	241 <code>\FamilyKeyStateProcessed%</code>
	242 <code>}{%</code>
	243 <code>\FamilySetLengthMacro{SVG}{width}{\svg@param@width}{#1}%</code>
	244 <code>\ifdim\svg@param@width<\z@\relax%</code>
	245 <code>\FamilyKeyStateUnknownValue%</code>
	246 <code>\fi%</code>
	247 <code>}%</code>
	248 <code>}</code>
	249 <code>\newcommand*\svg@param@height{\z@}</code>
	250 <code>\DefineFamilyKey{SVG}{height}{%</code>
	251 <code>\FamilyKeyStateUnknownValue%</code>
	252 <code>\svg@ifvalueisrelax{#1}{%</code>
	253 <code>\renewcommand*\svg@param@height{\z@}%</code>
	254 <code>\FamilyKeyStateProcessed%</code>
	255 <code>}{%</code>
	256 <code>\FamilySetLengthMacro{SVG}{height}{\svg@param@height}{#1}%</code>
	257 <code>\ifdim\svg@param@height<\z@\relax%</code>
	258 <code>\FamilyKeyStateUnknownValue%</code>
	259 <code>\fi%</code>
	260 <code>}%</code>
	261 <code>}</code>
	262 <code>\newcommand*\svg@param@scale{1}</code>
	263 <code>\DefineFamilyKey{SVG}{scale}{%</code>
	264 <code>\FamilyKeyStateUnknownValue%</code>
	265 <code>\svg@ifvalueisrelax{#1}{%</code>
	266 <code>\renewcommand*\svg@param@scale{1}%</code>
	267 <code>\FamilyKeyStateProcessed%</code>
	268 <code>}{%</code>
	269 <code>\ifisdimension{#1\p@}{%</code>
	270 <code>\ifdim\dimexpr#1\p@>\z@\relax%</code>
	271 <code>\renewcommand*\svg@param@scale{#1}%</code>
	272 <code>\FamilyKeyStateProcessed%</code>
	273 <code>\fi%</code>
	274 <code>}{}%</code>
	275 <code>}%</code>
	276 <code>}</code>
<code>pretex</code> (opt.)	For executing code right before or after the graphic inclusion, two hooks are defined.
<code>\svg@param@pretex</code>	277 <code>\newcommand*\svg@param@pretex{}</code>
<code>apptex</code> (opt.)	278 <code>\let\svg@param@pretex\relax</code>
<code>\svg@param@apptex</code>	279 <code>\DefineFamilyKey{SVG}{pretex}{%</code>
<code>postex</code> (opt.)	280 <code>\svg@ifvalueisrelax{#1}{%</code>
	281 <code>\let\svg@param@pretex\relax%</code>
	282 <code>}{%</code>

```

283 \def\svg@param@pretex{#1}%
284 }%
285 \FamilyKeyStateProcessed%
286 }
287 \newcommand*\svg@param@apptex{}
288 \let\svg@param@apptex\relax
289 \DefineFamilyKey{SVG}{apptex}{%
290 \svg@ifvalueisrelax{#1}{%
291 \let\svg@param@apptex\relax%
292 }{%
293 \def\svg@param@apptex{#1}%
294 }%
295 \FamilyKeyStateProcessed%
296 }
297 \DefineFamilyKey{SVG}{postex}{%
298 \svg@deprecated@key{postex=#1}{apptex=#1}%
299 }

```

`lastpage` (opt.)

`svg@param@lastpage` (counter)

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.

```

300 \newcounter{svg@param@lastpage}
301 \DefineFamilyKey{SVG}{lastpage}{%
302 \FamilySetNumerical{SVG}{lastpage}{svg@tempa}{%
303 {false}{0},{off}{0},{no}{0},{ignore}{0},%
304 {true}{1},{on}{1},{yes}{1},{auto}{1}%
305 }{#1}%
306 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
307 \ifcase\svg@tempa\relax% false
308 \FamilySetCounter{SVG}{lastpage}{svg@param@lastpage}{\m@ne}%
309 \or% true
310 \FamilySetCounter{SVG}{lastpage}{svg@param@lastpage}{\z@}%
311 \fi%
312 \fi%
313 }

```

`draft` (opt.)

`\if@svg@draft`

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

```

314 \newif\if@svg@draft
315 \FamilyBoolKey{SVG}{draft}{@svg@draft}
316 \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@quotes@remove`
`\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.

```

317 \newcommand*\svg@quotes@remove[1]{%

```

```

318 \begingroup%
319 \edef\svg@tempa{#1}%
320 \expandafter\svg@quotes@@remove\svg@tempa""\@nil%
321 \edef\svg@tempb{%
322 \endgroup%
323 \noexpand\def\noexpand#1{\unexpanded\expandafter{\svg@tempa}}%
324 }%
325 \svg@tempb%
326 }
327 \newcommand*\svg@quotes@@remove{}
328 \def\svg@quotes@@remove#1"#2"#3\@nil{%
329 \IfArgIsEmpty{#2}{%
330 \edef\svg@tempa{#1}%
331 }{%
332 \svg@quotes@@remove#1#2#3""\@nil%
333 }%
334 }

```

\svg@quotes@check 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.

```

335 \newif\if@svg@quotes@found
336 \newcommand*\svg@quotes@check[1]{%
337 \expandafter\svg@quotes@@check#1""\@nil%
338 }
339 \newcommand*\svg@quotes@@check{}
340 \def\svg@quotes@@check#1"#2\@nil{%
341 \IfArgIsEmpty{#2}{\@svg@quotes@foundfalse}{\@svg@quotes@foundtrue}%
342 }

```

\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.

```

343 \newcommand*\svg@normalize@path[1]{%
344 \begingroup%
345 \edef\svg@tempa{#1}%
346 \svg@quotes@check{\svg@tempa}%
347 \svg@quotes@remove{\svg@tempa}%
348 \ifx\svg@tempa\@empty\relax%
349 \def\svg@tempa{./}%
350 \fi%
351 \expandafter\svg@normalize@@path\svg@tempa//\@nil%
352 \edef\svg@tempb{%
353 \endgroup%
354 \if@svg@quotes@found%
355 \noexpand\def\noexpand#1{"\unexpanded\expandafter{\svg@tempa}}%
356 \else%
357 \noexpand\def\noexpand#1{\unexpanded\expandafter{\svg@tempa}}%
358 \fi%
359 }%
360 \svg@tempb%
361 }
362 \newcommand*\svg@normalize@@path{}
363 \def\svg@normalize@@path#1/#2/\@nil{%
364 \IfArgIsEmpty{#2}{%

```

```

365 \IfArgIsEmpty{#1}{\def\svg@tempa{}}{\def\svg@tempa{#1/}}%
366 }{%
367 \svg@normalize@@path#2/\@nil%
368 \edef\svg@tempa{#1/\unexpanded\expandafter{\svg@tempa}}%
369 }%
370 }

```

`\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.

```

371 \newcommand*\svg@ifvalueisrelax[1]{%
372 \begingroup%
373 \def\svg@tempa{#1}%
374 \def\svg@tempb{\relax}%
375 \ifx\svg@tempa\svg@tempb\relax%
376 \aftergroup\@firstoftwo%
377 \else%
378 \aftergroup\@secondoftwo%
379 \fi%
380 \endgroup%
381 }

```

`\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 `\svgpath` 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`.

`\svg@file@path`

`\svg@file@name`

`\svg@file@base`

`\svg@file@suffix`

```

382 \newif\if@svg@file@found
383 \newcommand*\svg@file@path{}
384 \newcommand*\svg@file@name{}
385 \newcommand*\svg@file@base{}
386 \newcommand*\svg@file@suffix{}
387 \newcommand*\svg@get@path[3][svg]{%
388 \begingroup%

```

A maybe given, unneeded file extension is removed.

```

389 \edef\svg@tempa{#2}%
390 \svg@quotes@check{\svg@tempa}%
391 \svg@quotes@remove{\svg@tempa}%
392 \expandafter\svg@filename@parse\expandafter{\svg@tempa}%
393 \IfArgIsEmpty{#1}{%
394 \edef\svg@tempa{\filename@area\filename@base.\filename@ext}%
395 }{%
396 \edef\svg@tempa{\filename@area\filename@base.#1}%
397 }%
398 \if@svg@quotes@found%
399 \edef\svg@tempa{"\svg@tempa"}%
400 \fi%

```

If `\svgpath` was used, it is searched first. 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.

```

401 \@svg@file@foundfalse%

```



```

402 \let\input@path\svg@path%
403 \svg@get@@path{\svg@tempa}%
404 \if@svg@file@found\else%
405 \ifx\Ginput@path\@undefined\else%
406 \let\input@path\Ginput@path%
407 \svg@get@@path{\svg@tempa}%
408 \fi%
409 \fi%
410 \IfArgIsEmpty{#3}{-}{%
411 \if@svg@file@found\else%
412 \ifx#3\@undefined\else%
413 \edef\svg@tempb{#{#3}}%
414 \let\input@path\svg@tempb%
415 \svg@get@@path{\svg@tempa}%
416 \fi%
417 \fi%
418 }%
419 \edef\svg@tempa{%
420 \endgroup%
421 \if@svg@file@found%
422 \noexpand\@svg@file@foundtrue%
423 \noexpand\def\noexpand\svg@file@path{\filename@area}%
424 \noexpand\def\noexpand\svg@file@name{\filename@base}%
425 \noexpand\def\noexpand\svg@file@base{\filename@area\filename@base}%
426 \else%
427 \noexpand\@svg@file@foundfalse%
428 \noexpand\def\noexpand\svg@file@path{}%
429 \noexpand\def\noexpand\svg@file@name{#2}%
430 \noexpand\def\noexpand\svg@file@base{#2}%
431 \fi%
432 }%
433 \svg@tempa%
434 }

```

The macro `\svg@get@@path` does the actual search job.

```

435 \newcommand*\svg@get@@path[1]{%
436 % The specified file is searched with \cs{IfFileExists}. If the file search was
437 % succesful, the macro \cs{svg@filename@parse} is called with the result.
438 % \begin{macrocode}
439 \expandafter\IfFileExists\expandafter{#1}{%
440 \@svg@file@foundtrue%
441 \expandafter\svg@filename@parse\expandafter{\@filef@und}%
442 }{}%
443 }

```

`\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.

```

444 \newcommand*\svg@filename@parse[1]{%
445 \begingroup%
446 \def\svg@tempa##1{%
447 \def\svg@tempb####1####2\@nil{%
448 \ifstr{####1}{"}{\def\svg@tempb{####2}}{\def\svg@tempb{####1####2}}%
449 }%
450 \expandafter\svg@tempb##1\@nil%
451 \edef##1{\svg@tempb}%
452 }%

```

The given path and file is parsed with `\filename@parse`. If an extension was found, it is appended to the file name for a second parsing run.

```

453 \filename@parse{#1}%
454 \ifx\filename@ext\relax\else%
455 \edef\filename@base{\filename@base.\filename@ext}%
456 \fi%
```

If there are quotes in the file path, the closing one will be found as first character in `\filename@base` as `\filename@area` is splitted at the last slash. This leading quote is removed from `\filename@base` with `\svg@tempa`.

```

457 \svg@quotes@check{\filename@area}%
458 \if@svg@quotes@found%
459 \svg@quotes@remove{\filename@area}%
460 \edef\filename@area{"\filename@area"}%
461 \svg@tempa{\filename@base}%
462 \fi%
```

Before the second call of `\filename@parse` remaining quotes are removed and the path in `\filename@area` is temporary stored in `\svg@tempa`.

```

463 \svg@quotes@check{\filename@base}%
464 \if@svg@quotes@found%
465 \svg@quotes@remove{\filename@base}%
466 \fi%
467 \let\svg@tempa\filename@area%
468 \expandafter\filename@parse\expandafter{\filename@base}%
469 \let\filename@area\svg@tempa%
470 \if@svg@quotes@found%
471 \edef\filename@base{"\filename@base"}%
472 \fi%
```

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

```

473 \edef\svg@tempa{%
474 \noexpand\endgroup%
475 \noexpand\def\noexpand\filename@area{\filename@area}%
476 \noexpand\def\noexpand\filename@base{\filename@base}%
477 \ifx\filename@ext\relax%
478 \noexpand\let\noexpand\filename@ext\noexpand\relax%
479 \else%
480 \noexpand\def\noexpand\filename@ext{\filename@ext}%
481 \fi%
482 }%
483 \svg@tempa%
484 }
```

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

```

485 \newcommand*\svg@file@missing[3][]{%
486 \begin{group}%
487 \edef\svg@tempa{#2}%
488 \expandafter\svg@filename@parse\expandafter{\svg@tempa}%
489 \svg@quotes@remove{\filename@area}%
490 \svg@quotes@remove{\filename@base}%
491 \ifx\filename@ext\relax\else%
492 \svg@quotes@remove{\filename@ext}%
493 \fi%
```

```

493 \fi%
494 \IfArgIsEmpty{#1}{%
495 \def\svg@tempa{%
496 Did you run the export with Inkscape? There's no file\MessageBreak%
497 '\filename@area\filename@base.\filename@ext'%
498 }%
499 }{%
500 \edef\filename@ext{#1}%
501 \edef\svg@tempb{#3}%
502 \ifstr{\svg@tempb}{.}{\let\svg@tempb\@empty}{}%
503 \ifstr{\filename@area}{.}{\let\filename@area\@empty}{}%
504 \def\svg@tempa{%
505 There's no file '\filename@base.\filename@ext'\MessageBreak%
506 \ifx\filename@area\@empty%
507 neither in the current directory nor\MessageBreak%
508 any other searched path given by\MessageBreak%
509 \string\svgpath%
510 \ifx\svg@path\@undefined\space\else%
511 \space(\svg@path)\MessageBreak%
512 \fi%
513 or \string\graphicspath%
514 \ifx\Ginput@path\@undefined\else%
515 \space(\Ginput@path)%
516 \fi%
517 \ifx\svg@tempb\@empty\else%
518 \MessageBreak or even 'inkscapepath' ('\svg@tempb')%
519 \fi.%
520 \else%
521 in folder '\filename@area'.%
522 \fi%
523 }%
524 }%
525 \PackageError{svg}{%
526 File '\filename@base.\filename@ext' is missing%
527 }{\svg@tempa}%
528 \endgroup%
529 }

```

`\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` is defined. Unfortunately this functionality isn't provided by XeTeX.

```

530 \ifx\pdf@filemoddate\@undefined
531 \newcommand*\svg@iffilenewer[2]{\@gobbletwo}
532 \else
533 \newcommand*\svg@iffilenewer[2]{%
534 \begingroup%
535 \edef\svg@tempa{\pdf@filemoddate{#1}}%
536 \edef\svg@tempb{\pdf@filemoddate{#2}}%
537 \ifnum\pdf@stricmp{\svg@tempa}{\svg@tempb}>\z@ \relax%
538 \aftergroup\@firstoftwo%
539 \else%
540 \aftergroup\@secondoftwo%
541 \fi%
542 \endgroup%
543 }
544 \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
`\svg@local@param@use` `\includeinkscape`. Some of them are overloaded for the usage as optional argument and
`\svg@local@param@def` 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**.

```
545 \newcommand*\svg@local@param@set[1]{%  
546   \svg@local@param@use%  
547   \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`.

```
548   \ifstr{\svg@ink@format}{png}{\FamilyOptions{SVG}{inkscapelatex=false}}{}%  
549 }  
550 \newcommand*\svg@local@param@use{}  
551 \newcommand*\svg@local@param@def[1]{%  
552   \edef\svg@local@param@use{%  
553     \unexpanded\expandafter{\svg@local@param@use}\unexpanded{#1}%  
554   }%  
555 }  
556 \DefineFamilyMember[.param]{SVG}
```

B.4. User commands

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

```
557 \newcommand*\svgsetup{\FamilyOptions{SVG}}  
558 \newcommand*\setsvg{\FamilyOptions{SVG}}
```

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

```
559 \newcommand*\svg@path{}  
560 \let\svg@path\input@path  
561 \newcommand*\svgpath[1]{%  
562   \def\svg@tempb{}%  
563   \@tfor\svg@tempa:=#1\do{%  
564     \ifx\svg@tempa\@empty\else%  
565       \svg@normalize@path{\svg@tempa}%  
566       \edef\svg@tempb{\svg@tempb\svg@tempa}%  
567     \fi%  
568   }%  
569   \ifx\svg@tempb\@empty\else%  
570     \let\svg@path\svg@tempb%  
571   \fi%  
572 }
```

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

```
573 \newcommand*{\includesvg}[2][]{%  
574   \begingroup%
```

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

```
575 \svg@deprecated@param%
```

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

```
inkscapearea (param.)
inkscapedpi (param.)
inkscapeopt (param.) 576 \svg@local@param@set{#1}%
```

`width` (param.)

`height` (param.)

`scale` (param.)

`pretex` (param.)

`apptex` (param.)

`draft` (param.)

`extract` (param.)

`extractpreamble` (param.)

`extractformat` (param.)

`extractwidth` (param.)

`extractheight` (param.)

`extractscale` (param.)

`extractpretex` (param.)

`extractapptex` (param.)

`extractruns` (param.)

`latexopt` (param.)

`convert` (param.)

`convertformat` (param.)

`convertdpi` (param.)

`magicksetting` (param.)

`magickoperator` (param.)

`gsop` (param.)

`gsdevice` (param.)

`clean` (param.)

`exclude` (param.)

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

```
577 \if@svg@ink@latex%
578 \def\svg@file@suffi{x}_{_svg-tex}%
579 \else%
580 \def\svg@file@suffi{x}_{_svg-raw}%
581 \fi%
582 \@onelevel@sanitize\svg@file@suffi{x}%
```

Searching all given paths for the relevant SVG file.

```
583 \svg@get@path{#2}{}%
584 \if@svg@file@found%
```

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

```
585 \svg@ink@run%
586 \IfFileExists{\svg@out@base}{}%
587 \@svg@file@foundfalse%
588 \svg@file@missing{\svg@out@base}{}%
589 }%
590 \if@svg@ink@latex%
591 \IfFileExists{\svg@out@base_tex}{}%
592 \@svg@file@foundfalse%
593 \svg@file@missing{\svg@out@base_tex}{}%
594 }%
595 \fi%
```

Include the resulting graphic file and maybe extract independent files.

```
596 \if@svg@file@found%
597 \svg@input{\svg@out@base}%
598 \svg@extract{\svg@out@base}%
599 \fi%
600 \else%
```

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

```
601 \svg@file@missing[svg]{\svg@file@base}{}%
602 \fi%
603 \endgroup%
604 }
```

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

```
605 \svg@local@param@def{%
606   \FamilyCounterKey[.param]{SVG}{lastpage}{svg@param@lastpage}%
607 }
```

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

```
608 \newcommand*\svg@param@angle{0}
609 \svg@local@param@def{%
610   \DefineFamilyKey[.param]{SVG}{angle}{%
611     \renewcommand*\svg@param@angle{#1}%
612     \FamilyKeyStateProcessed%
613   }%
614 }
615 \newcommand*\svg@param@origin{c}
616 \svg@local@param@def{%
617   \DefineFamilyKey[.param]{SVG}{origin}[c]{%
618     \renewcommand*\svg@param@origin{#1}%
619     \FamilyKeyStateProcessed%
620   }%
621 }
```

`\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.

```
622 \newcommand*{\includeinkscape}[2][{}]{%
623   \begin{group}
```

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

```
624   \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 `inkscapelatex` is set to `true` by default.

```
625   \filename@parse{#2}%
626   \ifx\filename@ext\relax\else%
627     \svg@quotes@remove{\filename@ext}%
628     \expandafter\lowercase\expandafter{%
629       \expandafter\def\expandafter\filename@ext\expandafter{\filename@ext}%
630     }%
631     \let\svg@tempb\filename@ext%
632     \def\svg@tempa##1_tex##2\@nil{\def\svg@tempb{##1}}%
633     \expandafter\svg@tempa\svg@tempb_tex\@nil%
634     \@for\svg@tempa:={pdf,eps,ps,png}\do{%
635       \ifstr{\svg@tempb}{\svg@tempa}{%
636         \edef\svg@tempa{%
637           \noexpand\FamilyOptions{SVG}{inkscapeformat=\svg@tempb}%
638         }%
639         \svg@tempa%
640       }{}%
641     }%
642     \ifstr{\filename@ext}{\svg@ink@format_tex}{%
643       \FamilyOptions{SVG}{inkscapelatex=true}%
644     }{}%
```


inkscapeformat (param.) All 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).

inkscapelatex (param.)

width (param.)

height (param.)

scale (param.) 646 \svg@local@param@set{#1}%

pretex (param.)

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

draft (param.)

lastpage (param.) 647 \expandafter\svg@get@path\expandafter[\svg@ink@format]{#2}{\svg@out@path}%

angle (param.) 648 \if@svg@file@found%

origin (param.) Checking the required files for graphic inclusion.

extract (param.)

extractpreamble (param.) 649 \edef\svg@out@name{\svg@file@name}%

extractformat (param.) 650 \edef\svg@out@base{\svg@file@path\svg@file@name.\svg@ink@format}%

extractwidth (param.) 651 \if@svg@ink@latex%

extractheight (param.) 652 \IfFileExists{\svg@out@base_tex}{%}{%

extractscale (param.) 653 \@svg@file@foundfalse%

extractpretex (param.) 654 \svg@file@missing{\svg@out@base_tex}{%}{%

extractapptex (param.) 655 }%

extractruns (param.) 656 \fi%

latexopt (param.) Include the resulting graphic file and maybe extract independent files.

convert (param.) 657 \if@svg@file@found%

convertformat (param.) 658 \svg@input{\svg@out@base}%

convertdpi (param.) 659 \svg@extract{\svg@out@base}%

magicksetting (param.) 660 \fi%

magickoperator (param.) 661 \else%

gsopt (param.)

gsdevice (param.)

clean (param.) 662 \svg@file@missing[\svg@ink@format]{\svg@file@base}{\svg@out@path}%

exclude (param.) 663 \fi%

664 \endgroup%

665 }

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

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.

```

666 \newcommand*\svg@deprecated@param{%
667   \@svg@tempswafalse%
668   \ifx\svgwidth\@undefined\else%
669     \edef\svg@tempa{\noexpand\FamilyOptions{SVG}{width=\svgwidth}}%
670     \svg@tempa%
671     \@svg@tempswatrue%
672   \fi%
673   \ifx\svgscale\@undefined\else%
674     \edef\svg@tempa{\noexpand\FamilyOptions{SVG}{scale=\svgscale}}%
675     \svg@tempa%
676     \@svg@tempswatrue%
677   \fi%
678   \if@svg@tempswa%
679     \PackageWarning{svg}{%

```

```

680     You should specify the image size with parameters\MessageBreak%
681     'width' and 'height' or 'scale' instead of using\MessageBreak%
682     '\string\svgscale' or '\string\svgwidth'%
683 }%
684 \let\svgwidth\@undefined%
685 \let\svgscale\@undefined%
686 \fi%
687 }

```

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

```

688 \newif\if@svg@ink@run
689 \newcommand*\svg@ink@run{%
690   \ifnum\svg@ink@mode>\z@\relax%
691     \beginingroup%

```

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.

```

692   \@svg@ink@runtrue%
693   \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.

```

694   \svg@iffilenewer{\svg@file@base.svg}{\svg@out@base}{}%
695   \@svg@ink@runfalse%
696   }%

```

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.

```

697   \if@svg@ink@latex%
698     \IfFileExists{\svg@out@base_tex}{%
699       \ifnum\pdf@shellescape=\@one\relax\if@svg@ink@run%
700         \svg@iffilenewer{\svg@out@base_tex}{\svg@out@base}{%
701           \@svg@ink@runfalse%
702           \edef\svg@tempa{\svg@out@base}%
703           \svg@quotes@remove{\svg@tempa}%
704           \PackageWarning{svg}{%
705             Since the encountered filedate of file\MessageBreak%
706             '\svg@tempa_tex' is newer than \MessageBreak%
707             '\svg@tempa' it's supposed that\MessageBreak%
708             you customized this file. To avoid an accidental\MessageBreak%
709             overwriting of this file, the Inkscape export\MessageBreak%
710             won't be done. If you want to overwrite the\MessageBreak%
711             existing file please choose the parameter\MessageBreak%
712             'inkscape=force'%
713           }%
714         }{}%
715       \fi\fi%
716     }\@svg@ink@runtrue}%
717   \fi%
718   \fi%

```

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

```

719      \if@svg@ink@run%
720      \ifnum\pdf@shellescape=\@one\relax%

```

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

```

721      \ifx\svg@ink@dpi\relax%
722      \ifstr{\svg@ink@format}{png}{%
723      \FamilyOptions{SVG}{inkscape=300}%
724      }{%
725      \fi%
726      \PackageInfo{svg}{%
727      Calling Inkscape%
728      \ifx\svg@ink@opt\empty\else%
729      \space with added options '\svg@ink@opt'%
730      \fi%
731      }%

```

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

```

732      \edef\svg@tempa{\svg@file@base}%
733      \edef\svg@tempb{\svg@out@name}%
734      \svg@quotes@remove{\svg@tempa}%
735      \svg@quotes@remove{\svg@tempb}%
736      \ShellEscape{\svg@ink@cmd{\svg@tempa}{\svg@tempb}}%
737      \IfFileExists{\svg@out@name.\svg@ink@format}{%
738      \edef\svg@tempb{\svg@tempb.\svg@ink@format}%
739      \svg@quotes@remove{\svg@out@base}%
740      \svg@shell@mkdir{\svg@out@path}%
741      \svg@shell@move{\svg@tempb}{\svg@out@base}%
742      \if@svg@ink@latex%
743      \svg@shell@move{\svg@tempb_tex}{\svg@out@base_tex}%
744      \fi%
745      }{%
746      \PackageWarning{svg}{%
747      The export with Inkscape failed for file\MessageBreak%
748      '\svg@tempa.svg'\MessageBreak%
749      Troubleshooting: Please check in the log file how\MessageBreak%
750      the invocation of Inkscape took place and try to\MessageBreak%
751      execute it yourself in the terminal%
752      }%
753      }%

```

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

```

754      \else%
755      \edef\svg@tempa{\svg@file@base}%
756      \svg@quotes@remove{\svg@tempa}%
757      \PackageWarning{svg}{%
758      You didn't enable 'shell escape' (or 'write18')\MessageBreak%
759      so it wasn't possible to launch the Inkscape export\MessageBreak%
760      for '\svg@tempa.svg'%
761      }%
762      \fi%
763      \fi%
764      \endgroup%
765      \fi%
766      }

```

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

```

767 \newcommand*\svg@ink@cmd[2]{%
768   \svg@ink@exe\space-z\space\svg@ink@area\space%
769   \ifx\svg@ink@dpi\relax\else--export-dpi=\svg@ink@dpi\space\fi%
770   \if\svg@ink@latex--export-latex\space\fi%
771   \svg@ink@opt\space%
772   --file="#1.svg"\space%
773   --export-\svg@ink@format="#2.\svg@ink@format"\space%
774 }
```

`\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.

```

775 \newcommand*\svg@get@lastpage[1]{%
776   \ifstr{\svg@ink@format}{pdf}{%
777     \begingroup%
778     \svg@tempcnt=\m@ne\relax%
779     \ifx\XeTeXpdfpagecount\undefined%
780       \ifpdf%
781         \ifx\pdfximage\undefined%
782           \ifx\saveimageresource\undefined\else%
783             \saveimageresource{#1}%
784             \svg@tempcnt=\lastsavedimageresourcepages\relax%
785           \fi%
786         \else%
787           \pdfximage{#1}%
788           \svg@tempcnt=\pdfastximagepages\relax%
789         \fi%
790       \fi%
791     \else%
792       \svg@tempcnt=\XeTeXpdfpagecount#1\relax%
793     \fi%
794     \ifnum\svg@tempcnt=\m@ne\relax%
795       \PackageWarning{svg}{%
796         It wasn't possible to detect the last page\MessageBreak%
797         of '#1'%
798       }%
799     \else%
800       \PackageInfo{svg}{Last page of '#1' is \the\svg@tempcnt}%
801     \fi%
802     \edef\svg@tempa{%
803       \noexpand\endgroup%
804       \noexpand\FamilyOptions{SVG}{lastpage=\the\svg@tempcnt}%
805     }%
806     \svg@tempa%
807   }{}%
808 }
```

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

```

809 \newcommand*\svg@wrn@scale{%
810   \ifdim\dimexpr\svg@param@scale\p@\relax=\p@\relax\else%
811     \@svg@tempswafalse%
812   \ifdim\svg@param@width>\z@\relax%
813     \@svg@tempswatrue%
```

```

814 \fi%
815 \ifdim\svg@param@height>\z@\relax%
816 \@svg@tempwattrue%
817 \fi%
818 \if@svg@tempswa%
819 \PackageWarning{svg}{%
820 The parameter 'scale' is only considered if neither\MessageBreak%
821 'width' nor 'height' are specified%
822 }%
823 \fi%
824 \fi%
825 }

```

`\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**.

```

826 \newcommand*\svg@input{\svg@@input}
827 \newcommand*\svg@@input[2][]{%
828 \IfArgIsEmpty{#1}{\svg@local@param@set{#1}}%
829 \if@svg@draft%
830 \@svg@ink@latexfalse%
831 \fi%

```

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`.

```

832 \edef\svg@tempa{#2}%
833 \if@svg@ink@latex%
834 \svg@patches{\svg@tempa}%
835 \ifnum\value{svg@param@lastpage}=\z@\relax%
836 \expandafter\svg@get@lastpage\expandafter{\svg@tempa}%
837 \fi%
838 \edef\svg@tempa{%
839 \ifx\svg@param@pretex\relax\else%
840 \noexpand\svg@param@pretex%
841 \fi%
842 \noexpand\input{\svg@tempa_tex}%
843 \ifx\svg@param@apptex\relax\else%
844 \noexpand\svg@param@apptex%
845 \fi%
846 }%

```

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

```

847 \ifdim\dimexpr\svg@param@angle\p@\relax=\z@\relax%
848 \svg@tempa%
849 \else%
850 \edef\svg@tempb{origin=\svg@param@origin}%
851 \expandafter\rotatebox\expandafter[\svg@tempb]{\svg@param@angle}{%
852 \svg@tempa%
853 }%
854 \fi%
855 \else%

```

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

```

856 \svg@wrn@scale%

```

```

857 \edef\svg@tempb{keepaspectratio,scale=\svg@param@scale}%
858 \ifdim\svg@param@height>\z@\relax%
859 \edef\svg@tempb{\svg@tempb,height=\svg@param@height}%
860 \fi%
861 \ifdim\svg@param@width>\z@\relax%
862 \edef\svg@tempb{\svg@tempb,width=\svg@param@width}%
863 \fi%
864 \ifdim\dimexpr\svg@param@angle\p@\relax=\z@\relax\else%
865 \edef\svg@tempb{%
866 \svg@tempb,origin=\svg@param@origin,angle=\svg@param@angle%
867 }%
868 \fi%
869 \if@svg@draft%
870 \edef\svg@tempb{\svg@tempb,draft}%
871 \else%
872 \edef\svg@tempb{\svg@tempb,draft=false}%
873 \fi%
874 \expandafter\includegraphics\expandafter[\svg@tempb]{\svg@tempa}%
875 \fi%
876 }

```

B.6. Patches

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

```

877 \newcommand*\svg@patches[1]{%
878 \let\svg@picture@saved\picture%
879 \let\picture\svg@picture@patched%
880 \let\svg@includegraphics@saved\includegraphics%
881 \let\includegraphics\svg@includegraphics@patched%
882 \edef\svg@includegraphics@file{#1}%
883 }

```

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

```

884 \newcommand*\svg@picture@saved{}
885 \newcommand*\svg@picture@patched{}
886 \newcommand*\svg@picture@patched{}
887 \long\def\svg@picture@patched#1{\svg@picture@patched@#1}
888 \def\svg@picture@patched@(#1,#2){%
889 \svg@wrn@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.

```

890 \ifdim\svg@param@height>\z@\relax%
891 \Gscale@div\svg@tempa{#1\p@}{#2\p@}%
892 \setlength\unitlength{\svg@param@height}%
893 \setlength\unitlength{\svg@tempa\unitlength}%
894 \ifdim\svg@param@width>\z@\relax%
895 \ifdim\unitlength>\svg@param@width\relax%
896 \setlength\unitlength{\svg@param@width}%

```

```

897     \fi%
898     \fi%
899     \else%

```

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

```

900     \ifdim\svg@param@width>\z@\relax%
901         \setlength\unitlength{\svg@param@width}%
902     \else%
903         \setlength\unitlength{\svg@param@scale\unitlength}%
904     \fi%
905     \fi%

```

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

```

906     \svg@picture@saved{#1,#2}%
907 }

```

```

\svg@includegraphics@saved
\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.

```

908 \DefineFamily{SVGpatch}
909 \DefineFamilyMember{SVGpatch}
910 \newcounter{svg@param@currpage}
911 \setcounter{svg@param@currpage}{\m@ne}
912 \FamilyCounterKey{SVGpatch}{page}{svg@param@currpage}
913 \DefineFamilyKey{SVGpatch}{width}{\FamilyKeyStateProcessed}
914 \newcommand*\svg@includegraphics@file{}
915 \newcommand*\svg@includegraphics@saved{}
916 \newcommand*\svg@includegraphics@patched[2][]{%
917     \FamilyOptions{SVGpatch}{#1}%

```

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

```

918     \ifnum\value{svg@param@lastpage}<\z@\relax%
919         \FamilySetCounter{SVGpatch}{page}{svg@param@currpage}{%
920             \the\value{svg@param@lastpage}%
921         }%
922     \fi%

```

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

```

923     \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.

```

924     \svg@includegraphics@saved[#{#1}]{\svg@includegraphics@file}%
925     \fi%
926 }

```

C. Extracting independent graphic files with package `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.

```
927 <*base>
928 \DefineFamilyMember[.dummy]{SVG}
929 \newcommand*\svg@dummy@key[2][]{%
930   \IfArgIsEmpty{#1}{%
931     \DefineFamilyKey[.dummy]{SVG}{#2}{%
932       \PackageWarning{svg}{%
933         The option key '#2' can only\MessageBreak%
934         be used with package 'svg-extract', but\MessageBreak%
935         you didn't load it%
936       }%
937       \FamilyKeyStateProcessed%
938     }%
939   }{%
940     \DefineFamilyKey[.dummy]{SVG}{#2}[{#1}]{%
941       \PackageWarning{svg}{%
942         The option key '#2' can only\MessageBreak%
943         be used with package 'svg-extract', but\MessageBreak%
944         you didn't load it%
945       }%
946       \FamilyKeyStateProcessed%
947     }%
948   }%
```

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

```
949   \BeforePackage{svg-extract}{\RelaxFamilyKey[.dummy]{SVG}{#2}}%
950 }
951 </base>
```

C.1.1. Controlling the extract process

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

`\if@svgx@run`

```
952 <*base>
953 \svg@dummy@key[true]{extract}
954 </base>
955 <*extract>
956 \newif\if@svgx@run
957 \DefineFamilyKey{SVG}{extract}[true]{%
958   \lowercase{\def\svg@tempa{#1}}%
959   \FamilySetNumerical{SVG}{extract}{svg@tempa}{%
960     {false}{0},{off}{0},{no}{0},%
961     {true}{1},{on}{1},{yes}{1},{onlynewer}{1},{newer}{1},%
962     {overwrite}{1},{force}{1},{forced}{1},%
963     {pdf}{2},{eps}{3},{ps}{4}}%
```



```

964 }{\svg@tempa}%
965 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
966 \ifcase\svg@tempa\relax% false
967 \@svgx@runfalse%
968 \or% true
969 \@svgx@runtrue%
970 \or% pdf
971 \FamilyOptions{SVG}{extractformat=pdf}%
972 \or% eps
973 \FamilyOptions{SVG}{extractformat=eps}%
974 \or% ps
975 \FamilyOptions{SVG}{extractformat=ps}%
976 \fi%
977 \fi%
978 }
979 </extract>

```

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

```

980 <*extract>
981 \DeclareOption{on}{\FamilyOptions{SVG}{extract=true}}
982 \DeclareOption{off}{\FamilyOptions{SVG}{extract=false}}
983 </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.)
984 <*base>
985 \svg@dumkey{extractformat}
986 \svg@dumkey[true]{pdf}
987 \svg@dumkey[true]{eps}
988 </base>
989 <*extract>
990 \newcommand*\svgx@format{pdf}
991 \ifxetex\else\ifpdf\else
992 \renewcommand*\svgx@format{eps}
993 \fi\fi
994 \DefineFamilyKey{SVG}{extractformat}{%
995 \lowercase{\edef\svgx@format{#1}}%
996 \FamilyKeyStateProcessed%
997 }
998 \DefineFamilyKey{SVG}{pdf}[true]{%
999 \FamilySetBool{SVG}{pdf}{\@svg@tempa}{#1}%
1000 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1001 \if@svg@tempa%
1002 \svgx@ifinlist{pdf}{\svgx@format}{}%
1003 \edef\svgx@format{\svgx@format,pdf}%
1004 }%
1005 \svg@deprecated@key{pdf}{extractformat={\svgx@format}}%
1006 \else%
1007 \FamilyKeyStateUnknownValue
1008 \fi%
1009 \fi%
1010 }
1011 \DefineFamilyKey{SVG}{eps}[true]{%
1012 \FamilySetBool{SVG}{eps}{\@svg@tempa}{#1}%
1013 \ifx\FamilyKeyState\FamilyKeyStateProcessed%

```

```

1014 \if@svg@tempwa%
1015 \svg@ifinlist{eps}{\svgx@format}{}%
1016 \edef\svgx@format{\svgx@format,eps}%
1017 }%
1018 \svg@deprecated@key{eps}{extractformat={\svgx@format}}%
1019 \else%
1020 \FamilyKeyStateUnknownValue
1021 \fi%
1022 \fi%
1023 }
1024 \end{extract}

```

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

```

\svgx@preamble
extractpreambleend (opt.)
end (opt.)
\svgx@endpreamble
1025 (*base)
1026 \svg@dummy@key{extractpreamble}
1027 \svg@dummy@key{preamble}
1028 \svg@dummy@key{extractpreambleend}
1029 \svg@dummy@key{end}
1030 \end{base}
1031 (*extract)
1032 \newcommand*\svgx@preamble{\jobname.\svgx@latex@ext}%
1033 \DefineFamilyKey{SVG}{extractpreamble}{%
1034 \renewcommand*\svgx@preamble{#1}%
1035 \FamilyKeyStateProcessed%
1036 }
1037 \DefineFamilyKey{SVG}{preamble}{%
1038 \svg@deprecated@key[svg-extract]{preamble=#1}{extractpreamble=#1}%
1039 }
1040 \newcommand*\svgx@endpreamble{}
1041 \expandafter\def\expandafter\svgx@endpreamble\expandafter{%
1042 \csname begin\endcsname{document}%
1043 }
1044 \DefineFamilyKey{SVG}{extractpreambleend}{%
1045 \renewcommand*\svgx@endpreamble{#1}%
1046 \FamilyKeyStateProcessed%
1047 }
1048 \DefineFamilyKey{SVG}{end}{%
1049 \svg@deprecated@key[svg-extract]{end=#1}{extractpreambleend=#1}%
1050 }
1051 \end{extract}

```

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

```

\svgx@runs (counter)
1052 (*base)
1053 \svg@dummy@key{extractruns}
1054 \end{base}
1055 (*extract)
1056 \newcounter{svgx@runs}
1057 \DefineFamilyKey{SVG}{extractruns}{%
1058 \FamilySetCounter{SVG}{extractruns}{svgx@runs}{#1}%
1059 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1060 \ifnum\value{svgx@runs}<\@ne\relax%
1061 \PackageWarning{svg-extract}{%
1062 The count for runs has to be at least one%
1063 }%
1064 \FamilySetCounter{SVG}{extractruns}{svgx@runs}{\@ne}%

```

```

1065 \fi%
1066 \fi%
1067 }
1068 </extract>

```

`latexexe` (opt.) The command and facultative options for the L^AT_EX call of the separate auxiliary file. The default is set according to the currently used compiler.

```

\svgx@latex@exe
  latexext (opt.)
\svgx@latex@ext
  latexopt (opt.)
\svgx@latex@opt
1069 < *base>
1070 \svg@dummy@key{latexexe}
1071 \svg@dummy@key{pdflatex}
1072 \svg@dummy@key{latexext}
1073 \svg@dummy@key{latexopt}
1074 < /base>
1075 < *extract>
1076 \ifxetex
1077 \newcommand*\svgx@latex@exe{xelatex}
1078 \else\ifluatex
1079 \ifpdf
1080 \newcommand*\svgx@latex@exe{lualatex}
1081 \else
1082 \newcommand*\svgx@latex@exe{lualatex --output-format=dvi}
1083 \fi
1084 \else\ifpdf
1085 \newcommand*\svgx@latex@exe{pdflatex}
1086 \else
1087 \newcommand*\svgx@latex@exe{latex}
1088 \fi\fi\fi
1089 \DefineFamilyKey{SVG}{latexexe}{%
1090 \renewcommand*\svgx@latex@exe{#1}%
1091 \FamilyKeyStateProcessed%
1092 }
1093 \DefineFamilyKey{SVG}{pdflatex}{%
1094 \svg@deprecated@key[svg-extract]{pdflatex=#1}{latexexe=#1}%
1095 }
1096 \newcommand*\svgx@latex@ext{tex}
1097 \DefineFamilyKey{SVG}{latexext}{%
1098 \renewcommand*\svgx@latex@ext{#1}%
1099 \FamilyKeyStateProcessed%
1100 }
1101 \newcommand*\svgx@latex@opt{}
1102 \DefineFamilyKey{SVG}{latexopt}{%
1103 \renewcommand*\svgx@latex@opt{#1}%
1104 \FamilyKeyStateProcessed%
1105 }
1106 < /extract>

```

`dvipsopt` (opt.) Options and macros for calling convert commands, which are supplied by most L^AT_EX 2_ε distributions. These are used to generate all files, which are supported by option `extractformat`, as they don't need an additional application.

```

\svgx@dvips@exe
\svgx@dvips@opt
  pstoept (opt.)
\svgx@pstoept@exe
\svgx@pstoept@opt
  pstopdfopt (opt.)
\svgx@pstopdf@exe
\svgx@pstopdf@opt
  pdftoept (opt.)
\svgx@pdftoept@exe
\svgx@pdftoept@opt
  pdftops (opt.)
\svgx@pdftops@exe
\svgx@pdftops@opt
  pdftops (opt.)

```

```

1114 </base>
1115 <*extract>
1116 \newcommand*\svgx@dvi@exe{dvips}
1117 \newcommand*\svgx@dvi@opt{}
1118 \DefineFamilyKey{SVG}{dvi@opt}{%
1119   \renewcommand*\svgx@dvi@opt{#1}%
1120   \FamilyKeyStateProcessed%
1121 }
1122 \newcommand*\svgx@psto@exe{ps2eps}
1123 \newcommand*\svgx@psto@opt{-B -C}
1124 \DefineFamilyKey{SVG}{psto@opt}{%
1125   \renewcommand*\svgx@psto@opt{#1}%
1126   \FamilyKeyStateProcessed%
1127 }
1128 \newcommand*\svgx@pstopdf@exe{ps2pdf}
1129 \newcommand*\svgx@pstopdf@opt{}
1130 \DefineFamilyKey{SVG}{pstopdf@opt}{%
1131   \renewcommand*\svgx@pstopdf@opt{#1}%
1132   \FamilyKeyStateProcessed%
1133 }
1134 \newcommand*\svgx@pdfto@exe{pdftops -eps}
1135 \newcommand*\svgx@pdfto@opt{}
1136 \DefineFamilyKey{SVG}{pdfto@opt}{%
1137   \renewcommand*\svgx@pdfto@opt{#1}%
1138   \FamilyKeyStateProcessed%
1139 }
1140 \newcommand*\svgx@pdftops@exe{pdftops}
1141 \newcommand*\svgx@pdftops@opt{}
1142 \DefineFamilyKey{SVG}{pdftops@opt}{%
1143   \renewcommand*\svgx@pdftops@opt{#1}%
1144   \FamilyKeyStateProcessed%
1145 }
1146 \DefineFamilyKey{SVG}{pdftops}{%
1147   \PackageWarning{#1}{%
1148     The option key 'pdftops' is deprecated.\MessageBreak%
1149     You should use either 'pdfto@opt' or\MessageBreak%
1150     'pdftops@opt' instead. See the manual for\MessageBreak%
1151     more. Nothing was done%
1152   }%
1153   \FamilyKeyStateProcessed
1154 }
1155 </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. *ImageMagick* is set by default.

```

\if@svgx@cnv@run
\svgx@cnv@cmd

```

```

1156 <*base>
1157 \svg@dum@key[true]{convert}
1158 </base>
1159 <*extract>
1160 \newif\if@svgx@cnv@run
1161 \newcommand*\svgx@cnv@cmd{}

```

```

1162 \DefineFamilyKey{SVG}{convert}[true]{%
1163   \FamilySetNumerical{SVG}{convert}{svg@tempa}{%
1164     {false}{0},{off}{0},{no}{0},%
1165     {true}{1},{on}{1},{yes}{1},{onlynewer}{1},{newer}{1},%
1166     {overwrite}{1},{force}{1},{forced}{1},%
1167     {magick}{2},{imagemagick}{2},{convert}{2},%
1168     {gs}{3},{ghostscript}{3},%
1169     {gs64}{4},{ghostscript64}{4},%
1170     {gs32}{5},{ghostscript32}{5}%
1171   }{#1}%
1172   \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1173     \ifcase\svg@tempa\relax% false
1174       \@svgx@cnv@runfalse%
1175     \or% true
1176       \@svgx@cnv@runtrue%
1177     \or% magick
1178       \@svgx@cnv@runtrue%
1179       \renewcommand*\svgx@cnv@cmd{\svgx@magick@cmd}%
1180     \or% gs
1181       \@svgx@cnv@runtrue%
1182       \renewcommand*\svgx@cnv@cmd{\svgx@gs@cmd}%
1183     \or% gs64
1184       \@svgx@cnv@runtrue%
1185       \renewcommand*\svgx@cnv@cmd{\svgx@gs@cmd}%
1186       \svgx@onlywindows{%
1187         \renewcommand*\svgx@gs@exe{gswin64c}%
1188       }%
1189     \or% gs32
1190       \@svgx@cnv@runtrue%
1191       \renewcommand*\svgx@cnv@cmd{\svgx@gs@cmd}%
1192       \svgx@onlywindows{%
1193         \renewcommand*\svgx@gs@exe{gswin32c}%
1194       }%
1195     \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.

```

1196   \else%

```

Same doing like with option `inkscape`.

```

1197   \def\svg@tempa##1-##2\@nil{%
1198     \IfArgIsEmpty{##2}{\def\svg@tempb{}}{%
1199       \def\svg@tempa##1####1\@nil{\def\svg@tempb{####1}}%
1200       \svg@tempa#1\@nil%
1201     }%
1202     \def\svg@tempa{##1}%
1203   }%
1204   \svg@tempa#1-\@nil%
1205   \PackageWarning{svg-extract}{%
1206     Setting the executable%
1207     \ifx\svg@tempb\@empty\else%
1208       \space and associated options%
1209     \fi%
1210   \MessageBreak%
1211   for ImageMagick should be done with options\MessageBreak%
1212   'magickexe=\svg@tempa'%

```

```

1213 \ifx\svg@tempb\@empty\else%
1214 \MessageBreak and ‘magicksetting’ and/or ‘magickoperator’%
1215 \fi.\MessageBreak%
1216 Nevertheless, this was done by now%
1217 \ifx\svg@tempb\@empty\else%
1218 , whereby \MessageBreak ‘magicksetting=\svg@tempb’ was used%
1219 \fi%
1220 }%
1221 \FamilyOptions{SVG}{convert=magick}%
1222 \edef\svg@tempa{%
1223 \noexpand\FamilyOptions{SVG}{magickexe=\svg@tempa}%
1224 \ifx\svg@tempb\@empty\else%
1225 \noexpand\FamilyOptions{SVG}{magicksetting=\svg@tempb}%
1226 \fi%
1227 }%
1228 \svg@tempa%
1229 \fi%
1230 }
1231 \extract

```

`convertformat` (opt.) Option `convertformat` controls the output format for converted files. It is set to `png` by default.

`\svgx@cnv@format`

`png` (opt.)

```

1232 (*base)
1233 \svg@dumy@key{convertformat}
1234 \svg@dumy@key[true]{png}
1235 \base)
1236 (*extract)
1237 \newcommand*\svgx@cnv@format{png}
1238 \DefineFamilyKey{SVG}{convertformat}{%
1239 \lowercase{\edef\svgx@cnv@format{#1}}%
1240 \ifx\svgx@cnv@format\@empty\else%
1241 \@svgx@cnv@runtrue%
1242 \fi%
1243 \FamilyKeyStateProcessed%
1244 }
1245 \DefineFamilyKey{SVG}{png}[true]{%
1246 \FamilySetBool{SVG}{png}{\svg@tempswa}{#1}%
1247 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1248 \if\svg@tempswa%
1249 \svgx@ifinlist{png}{\svgx@cnv@format}{}%
1250 \edef\svgx@cnv@format{\svgx@cnv@format,png}%
1251 }%
1252 \svg@deprecated@key{png}{convertformat={\svgx@cnv@format}}%
1253 \else%
1254 \FamilyKeyStateUnknownValue
1255 \fi%
1256 \fi%
1257 }
1258 \extract

```

`convertdpi` (opt.) The option `convertdpi` is meant to define the used density during the conversion process. It can be set either for all designated output formats or targeted for a specific format. It’s also 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`.

`\svgx@cnv@dpi`

```

1259 (*base)

```

```

1260 \svg@dummy@key{convertdpi}
1261 \svg@dummy@key{convertdensity}
1262 </base>
1263 <*extract>
1264 \newcommand*\svgx@cnv@dpi{}
1265 \let\svgx@cnv@dpi\relax
1266 \DefineFamilyKey{SVG}{convertdpi}{%
1267   \FamilyKeyStateUnknownValue%
1268   \svgx@ifkeyandval{#1}{%
1269     \svgx@cnv@get@dpi{##2}%
1270     \ifx\svg@tempa\relax\else%
1271       \expandafter\edef\csname svgx@cnv@dpi@##1\endcsname{\svg@tempa}%
1272       \FamilyKeyStateProcessed%
1273     \fi%
1274   }{%
1275     \svgx@cnv@get@dpi{##1}%
1276     \ifx\svg@tempa\relax\else%
1277       \edef\svgx@cnv@dpi{\svg@tempa}%
1278       \FamilyKeyStateProcessed%
1279     \fi%
1280   }%
1281 }
1282 \DefineFamilyKey{SVG}{convertdensity}{\FamilyOptions{SVG}{convertdpi=#1}}
1283 </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
1284 <*base>
1285 \svg@dummy@key{magickexe}
1286 \svg@dummy@key{magicksetting}
1287 \svg@dummy@key{magickoperator}
1288 </base>
1289 <*extract>
1290 \newcommand*\svgx@magick@exe{}
1291 \DefineFamilyKey{SVG}{magickexe}{%
1292   \renewcommand*\svgx@magick@exe{#1}%
1293   \FamilyKeyStateProcessed%
1294 }
1295 \newcommand*\svgx@magick@set{}
1296 \DefineFamilyKey{SVG}{magicksetting}{%
1297   \svgx@setformatkey{#1}{\svgx@magick@set}%
1298   \FamilyKeyStateProcessed%
1299 }
1300 \newcommand*\svgx@magick@opr{}
1301 \DefineFamilyKey{SVG}{magickoperator}{%
1302   \svgx@setformatkey{#1}{\svgx@magick@opr}%
1303   \FamilyKeyStateProcessed%
1304 }
1305 </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**.

gsdevice (opt.)

\svgx@gs@exe

gsopt (opt.)

\svgx@gs@opt

gsdevice (opt.)

\svgx@gs@device

```

1306 (*base)
1307 \svg@dummy@key{gsexex}
1308 \svg@dummy@key{gsopt}
1309 \svg@dummy@key{gsdevice}
1310 (/base)
1311 (*extract)
1312 \newcommand*\svgx@gs@exe{}
1313 \DefineFamilyKey{SVG}{gsexex}{%
1314   \renewcommand*\svgx@gs@exe{#1}%
1315   \FamilyKeyStateProcessed%
1316 }
1317 \newcommand*\svgx@gs@opt{}
1318 \DefineFamilyKey{SVG}{gsopt}{%
1319   \svgx@setformatkey{#1}{svgx@gs@opt}%
1320   \FamilyKeyStateProcessed%
1321 }
1322 \newcommand*\svgx@gs@device{}
1323 \DefineFamilyKey{SVG}{gsdevice}{%
1324   \svgx@setformatkey{#1}{svgx@gs@device}%
1325   \FamilyKeyStateProcessed%
1326 }
1327 (/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.

```

name (opt.)
\svgx@out@path
\svgx@out@name
\if@svgx@out@sec
svgx@out@count (counter)
1328 (*base)
1329 \svg@dummy@key{extractpath}
1330 \svg@dummy@key{path}
1331 \svg@dummy@key{extractname}
1332 \svg@dummy@key{name}
1333 (/base)
1334 (*extract)
1335 \newcommand*\svgx@out@path{}
1336 \DefineFamilyKey{SVG}{extractpath}{%
1337   \FamilySetNumerical{SVG}{extractpath}{svg@tempa}{%
1338     {svgpath}{0},{svgdir}{0},%
1339     {svgsubpath}{1},{svgsubdir}{1},%
1340     {basepath}{2},{basedir}{2},{jobpath}{2},{jobdir}{2},%
1341     {basesubpath}{3},{basesubdir}{3},{jobsubpath}{3},{jobsubdir}{3}%
1342   }{#1}%
1343   \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1344     \ifcase\svg@tempa\relax% svgpath
1345       \renewcommand*\svgx@out@path{\svg@file@path}%
1346     \or% svgsubpath
1347       \renewcommand*\svgx@out@path{\svg@file@path svg-extract/}%
1348     \or% basepath
1349       \renewcommand*\svgx@out@path{./}%
1350     \or% basesubpath
1351       \renewcommand*\svgx@out@path{./svg-extract/}%
1352     \fi%
1353   \else%
1354     \renewcommand*\svgx@out@path{#1}%
1355     \svg@normalize@path{\svgx@out@path}%
1356     \FamilyKeyStateProcessed%

```



```

1357 \fi%
1358 }
1359 \DefineFamilyKey{SVG}{path}{%
1360 \svg@deprecated@key[svg-extract]{path=#1}{extractpath=#1}%
1361 }
1362 \newcounter{svgx@out@count}
1363 \newcommand*\svgx@out@name{}
1364 \newif\if@svgx@out@sec
1365 \DefineFamilyKey{SVG}{extractname}{%
1366 \FamilySetNumerical{SVG}{extractname}{svg@tempa}{%
1367 {filename}{0},{name}{0},%
1368 {filenamenumbered}{1},{namenumbered}{1},%
1369 {numberedfilename}{1},{numberedname}{1},%
1370 {numbered}{2},{section}{2},{numberedsection}{2},{sectionnumbered}{2}%
1371 }{#1}%
1372 \@svgx@out@secfalse%
1373 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1374 \ifcase\svg@tempa\relax% filename
1375 \renewcommand*\svgx@out@name{\svg@out@name-extract}%
1376 \or% filenamenumbered
1377 \renewcommand*\svgx@out@name{\the\value{svgx@out@count}-\svg@out@name}%
1378 \or% numbered
1379 \renewcommand*\svgx@out@name{\the\value{svgx@out@count}-\svgx@out@sec}%
1380 \@svgx@out@sectrue%
1381 \fi%
1382 \else%
1383 \def\svg@tempa##1.##2\@nil{%
1384 \IfArgIsEmpty{##1}{\renewcommand*\svgx@out@name{##1}}%
1385 }%
1386 \svg@tempa#1.\@nil%
1387 \FamilyKeyStateProcessed%
1388 \fi%
1389 }
1390 \DefineFamilyKey{SVG}{name}{%
1391 \svg@deprecated@key[svg-extract]{name=#1}{extractname=#1}%
1392 }
1393 \extract

```

C.1.4. Options for the extraction of graphics

`extractwidth` (opt.) For graphic extraction, the given settings regarding the size for inclusion can be overwritten with these options. Using `\relax` as value leads to resetting an option as unset, regardless of what was previously given. The value `inherit` means, that the actual option for including is used for extraction as well. This is the default setting.

```

\svgx@param@width
extractheight (opt.)
\svgx@param@width
extractscale (opt.)
\svgx@param@scale
1394 (*base)
1395 \svg@dummy@key{extractwidth}
1396 \svg@dummy@key{extractheight}
1397 \svg@dummy@key{extractscale}
1398 \base
1399 (*extract)
1400 \newcommand*\svgx@param@width{\svg@param@width}
1401 \DefineFamilyKey{SVG}{extractwidth}{%
1402 \FamilyKeyStateUnknownValue%
1403 \svg@ifvalueisrelax{#1}{%
1404 \renewcommand*\svgx@param@width{\z@}%
1405 \FamilyKeyStateProcessed%
1406 }{

```

```

1407 \ifstr{#1}{inherit}{%
1408 \renewcommand*\svgx@param@width{\svg@param@width}%
1409 \FamilyKeyStateProcessed%
1410 }{%
1411 \FamilySetLengthMacro{SVG}{extractwidth}{\svg@param@width}{#1}%
1412 \ifdim\svgx@param@width<\z@ \relax%
1413 \FamilyKeyStateUnknownValue%
1414 \fi%
1415 }%
1416 }%
1417 }
1418 \newcommand*\svgx@param@height{\svg@param@height}
1419 \DefineFamilyKey{SVG}{extractheight}{%
1420 \FamilyKeyStateUnknownValue%
1421 \svg@ifvalueisrelax{#1}{%
1422 \renewcommand*\svgx@param@height{\z@}%
1423 \FamilyKeyStateProcessed%
1424 }{%
1425 \ifstr{#1}{inherit}{%
1426 \renewcommand*\svgx@param@height{\svg@param@height}%
1427 \FamilyKeyStateProcessed%
1428 }{%
1429 \FamilySetLengthMacro{SVG}{extractheight}{\svg@param@height}{#1}%
1430 \ifdim\svgx@param@height<\z@ \relax%
1431 \FamilyKeyStateUnknownValue%
1432 \fi%
1433 }%
1434 }%
1435 }
1436 \newcommand*\svgx@param@scale{\svg@param@scale}
1437 \DefineFamilyKey{SVG}{extractscale}{%
1438 \FamilyKeyStateUnknownValue%
1439 \svg@ifvalueisrelax{#1}{%
1440 \renewcommand*\svgx@param@scale{1}%
1441 \FamilyKeyStateProcessed%
1442 }{%
1443 \ifstr{#1}{inherit}{%
1444 \renewcommand*\svgx@param@scale{\svg@param@scale}%
1445 \FamilyKeyStateProcessed%
1446 }{%
1447 \ifisdimension{#1\p@}{%
1448 \ifdim\dimexpr#1\p@ \relax > \z@ \relax%
1449 \renewcommand*\svgx@param@scale{#1}%
1450 \FamilyKeyStateProcessed%
1451 \fi%
1452 }{}%
1453 }%
1454 }%
1455 }
1456 </extract>

```

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

`\svgx@param@pretex`

`extractapptex` (opt.)

`\svgx@param@apptex`

`extractpostex` (opt.)

```

1457 (*base)
1458 \svg@dummys@key{extractpretex}
1459 \svg@dummys@key{extractapptex}
1460 \svg@dummys@key{extractpostex}
1461 </base>
1462 (*extract)

```

```

1463 \newcommand*\svgx@param@pretex{\svg@param@pretex}
1464 \DefineFamilyKey{SVG}{extractpretex}{%
1465   \svg@ifvalueisrelax{#1}{%
1466     \let\svgx@param@pretex\relax%
1467   }{%
1468     \ifstr{#1}{inherit}{%
1469       \def\svgx@param@pretex{\svg@param@pretex}%
1470     }{%
1471       \def\svgx@param@pretex{#1}%
1472     }%
1473   }%
1474   \FamilyKeyStateProcessed%
1475 }
1476 \newcommand*\svgx@param@apptex{\svg@param@apptex}
1477 \DefineFamilyKey{SVG}{extractapptex}{%
1478   \svg@ifvalueisrelax{#1}{%
1479     \let\svgx@param@apptex\relax%
1480   }{%
1481     \ifstr{#1}{inherit}{%
1482       \def\svgx@param@apptex{\svg@param@apptex}%
1483     }{%
1484       \def\svgx@param@apptex{#1}%
1485     }%
1486   }%
1487   \FamilyKeyStateProcessed%
1488 }
1489 \DefineFamilyKey{SVG}{extractpostex}{%
1490   \svg@deprecated@key[svg-extract]{extractpostex=#1}{extractapptex=#1}%
1491 }
1492 </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 be given.

`\svgx@clean`

```

1493 <*base>
1494 \svg@dummy@key[true]{clean}
1495 \svg@dummy@key[true]{clear}
1496 </base>
1497 <*extract>
1498 \newcommand*\svgx@clean{}
1499 \DefineFamilyKey{SVG}{clean}[true]{%
1500   \FamilySetBool{SVG}{clean}{@svg@tempswa}{#1}%
1501   \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1502     \if@svg@tempswa%
1503       \renewcommand*\svgx@clean{log,aux,dvi,out,ps,eps,pdf,\svgx@latex@ext}%
1504     \else%
1505       \renewcommand*\svgx@clean{}%
1506     \fi%
1507   \else%
1508     \renewcommand*\svgx@clean{#1}%
1509     \FamilyKeyStateProcessed%
1510   \fi%
1511 }
1512 \DefineFamilyKey{SVG}{clear}{\FamilyOptions{SVG}{clean=#1}}
1513 </extract>

```

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

```

1514 <*base>
1515 \svg@dummys@key[true]{exclude}
1516 </base>
1517 <*extract>
1518 \DefineFamilyKey{SVG}{exclude}[true]{%
1519   \FamilySetBool{SVG}{exclude}{@svg@tempswa}{#1}%
1520   \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1521     \if@svg@tempswa%
1522       \renewcommand*\svg@input[2][]{%
1523         \if@svgx@run\else%
1524           \PackageWarning{svg-extract}{%
1525             The image ‘##2’ was\MessageBreak%
1526             neither extracted nor included%
1527           }%
1528         \fi%
1529       }%
1530     \else%
1531       \renewcommand*\svg@input{\svg@@input}%
1532     \fi%
1533   \fi%
1534 }
1535 </extract>

```

C.2. User commands

\svghidepreamblestart Some dummies for package **svg**.
\svghidepreambleend

```

1536 <*base>
1537 \newcommand*\svghidepreamblestart{%
1538   \PackageWarning{svg}{%
1539     The macro ‘\string\svghidepreamblestart’ is only meant\MessageBreak%
1540     to be used together with package ‘svg-extract’.\MessageBreak%
1541     Nevertheless, nothing will happen%
1542   }%
1543 }
1544 \newcommand*\svghidepreambleend{%
1545   \PackageWarning{svg}{%
1546     The macro ‘\string\svghidepreambleend’ is only meant\MessageBreak%
1547     to be used together with package ‘svg-extract’.\MessageBreak%
1548     Nevertheless, nothing will happen%
1549   }%
1550 }
1551 </base>

```

These two macros can be used to hide some parts of the preamble during reading the preamble of the main document.

```

1552 <*extract>
1553 \let\svghidepreamblestart\relax
1554 \let\svghidepreambleend\relax
1555 </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
1556 (*base)
1557 \newcommand*\svg@extract[1]{%
1558 /base)
1559 (*extract)
1560 \ifnum\pdf@shellescape=\@ne\relax\else%
1561 \renewcommand*\svg@extract[1]{%
1562 \if@svgx@run%
1563 \begingroup%
1564 \edef\svg@tempa{#1}%
1565 \svg@quotes@remove{\svg@tempa}%
1566 \PackageWarning{svg-extract}{%
1567 You didn't enable 'shell escape' (or 'write18')\MessageBreak%
1568 so it wasn't possible to run the extraction for\MessageBreak%
1569 file '\svg@tempa'%
1570 }%
1571 \endgroup%
1572 \fi%
1573 }%
1574 \expandafter\endinput%
1575 \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.

```

1576 \newread\svgx@stream@in
1577 \newcommand*\svgx@read@line{}
1578 \newwrite\svgx@stream@out
1579 \newif\if@svgx@preamble@write
1580 \renewcommand*\svg@extract[1]{%

```

If option `extract` is enabled...

```

1581 \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.

```

1582 \if@svgx@out@sec%
1583 \svgx@get@out@sec%
1584 \fi%
1585 \stepcounter{svgx@out@count}%
1586 \begingroup%
1587 \def\svg@tempa##1.##2\@nil{%
1588 \IfArgIsEmpty{##2}{\edef\svgx@preamble{##1.\svgx@latex@ext}}{}%
1589 }%
1590 \expandafter\svg@tempa\svgx@preamble.\@nil%
1591 \IfFileExists{\svg@file@path\svgx@preamble}{%
1592 \@svg@file@foundtrue%
1593 }{%

```

```

1594 \svg@get@path[]{\svgx@preamble}{\svg@out@path}%
1595 \def\svg@tempa####1.####2\@nil{%
1596 \edef\svgx@preamble{\svg@file@name.####2}%
1597 }%
1598 \expandafter\svg@tempa\svgx@preamble\@nil%
1599 }%
1600 \edef\svg@tempa{%
1601 \endgroup%
1602 \if@svg@file@found%
1603 \ifx\svg@file@path\@empty%
1604 \def\noexpand\svgx@preamble{./\svgx@preamble}%
1605 \else%
1606 \def\noexpand\svgx@preamble{\svg@file@path\svgx@preamble}%
1607 \fi%
1608 \fi%
1609 }%
1610 \svg@tempa%
1611 \beginingroup%
1612 \endlinechar=\m@ne%
1613 \IfFileExists{\svgx@preamble}{%
1614 \PackageInfo{svg-extract}{%
1615 The preamble file '\svgx@preamble'\MessageBreak%
1616 is used for the generation of the auxiliary file\MessageBreak%
1617 '\svgx@out@name.\svgx@latex@ext'%
1618 }%

```

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

```

1619 \catcode'\#=12\relax%
1620 \immediate\openout\svgx@stream@out=\svgx@out@name.\svgx@latex@ext%
1621 \immediate\openin\svgx@stream@in=\svgx@preamble%
1622 \@svg@tempswatrue%
1623 \@svgx@preamble@writetrue%
1624 \def\svgx@read@line{%

```

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

```

1625 \@whiles\if@svg@tempswa\fi{%
1626 \immediate\read\svgx@stream@in to\svgx@read@line%
1627 \ifx\svgx@read@line\@empty%
1628 \ifeof\svgx@stream@in\@svg@tempswafalse\fi%
1629 \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`

```

1630 \svgx@read@preamble@till{\svghidepreamblestart}{}%
1631 \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.

```

1632 \svgx@read@preamble@till{\svgx@endpreamble}{\@svg@tempswafalse}%
1633 \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.

```

1634         \if@svgx@classfound\else%
1635         \expandafter\svgx@documentclass%
1636         \svgx@read@line\documentclass\documentclass\@nil%
1637         \fi%

```

Writing out the—maybe manipulated—read in line.

```

1638         \ifx\svgx@read@line\@empty\else%
1639         \immediate\write\svgx@stream@out{%
1640         \unexpanded\expandafter{\svgx@read@line}%
1641         }%
1642         \fi%
1643         \fi%
1644         \fi%
1645     }%
1646     \immediate\closein\svgx@stream@in%
1647     \immediate\closeout\svgx@stream@out%
1648     \catcode'\#6\relax%

```

Once the separate auxiliary \LaTeX 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.

```

1649     \immediate\openin\svgx@stream@in=\svgx@out@name.\svgx@latex@ext%
1650     \def\svg@tempa{}%
1651     \loop\unless\ifeof\svgx@stream@in%
1652         \readline\svgx@stream@in to\svgx@read@line%
1653         \ifx\svgx@read@line\@empty\else%
1654             \edef\svg@tempa{%
1655                 \unexpanded\expandafter{\svg@tempa}%
1656                 \unexpanded\expandafter{\svgx@read@line}^^J%
1657             }%
1658             \fi%
1659         \repeat%
1660     \immediate\closein\svgx@stream@in%
1661 }{%

```

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

```

1662     \svg@quotes@remove{\svgx@preamble}%
1663     \ifx\svgx@preamble\@empty\else%
1664         \PackageWarning{svg-extract}{%
1665             The preamble file '\svgx@preamble'\MessageBreak%
1666             does not exist%
1667         }%
1668     \fi%
1669     \def\svg@tempa{}%
1670 }%

```

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

```

1671     \immediate\openout\svgx@stream@out=\svgx@out@name.\svgx@latex@ext%
1672     \immediate\write\svgx@stream@out{%
1673         \@percentchar\@percentchar\space This file was generated by package
1674         'svg-extract'^^J%
1675         \@percentchar\@percentchar\space from source '\jobname'^^J%

```

```

1676      \@percentchar\@percentchar\space It's intended to be compiled with
1677      '\svgx@latex@exe\ifx\svgx@latex@opt\@empty\else\space\svgx@latex@opt\fi'
1678      }%

```

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.

```

1679      \immediate\write\svgx@stream@out{%
1680      \string\AtBeginDocument{\@percentchar^^J%
1681      \space\space\string\svgxsetpapersize\@percentchar^^J%
1682      \ifxetex\else\ifpdf\else%
1683      \space\space\string\AtBeginDvi{\string\special{%
1684      papersize=\string\the\string\paperwidth,%
1685      \string\the\string\paperheight%
1686      }}\@percentchar^^J%
1687      \fi\fi%
1688      }^^J%
1689      \string\PassOptionsToPackage{hidelinks}{hyperref}%
1690      }%

```

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

```

1691      \if@svgx@classfound\else%
1692      \immediate\write\svgx@stream@out{\string\documentclass{article}}}%
1693      \fi%

```

And now the stored preamble.

```

1694      \ifx\svg@tempa\@empty\else%
1695      \immediate\write\svgx@stream@out{\unexpanded\expandafter{\svg@tempa}}}%
1696      \fi%

```

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

```

1697      \immediate\write\svgx@stream@out{\string\usepackage{svg-extract}}}%

```

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

```

1698      \def\svg@tempa##1{%
1699      \immediate\write\svgx@stream@out{\string\svgsetup{##1}}}%
1700      }%
1701      \if@svg@ink@latex\else%
1702      \svg@tempa{inkscapelatex=false}%
1703      \fi%
1704      \ifdim\svgx@param@width>\z@\relax%
1705      \svg@tempa{width=\svgx@param@width}%
1706      \fi%
1707      \ifdim\svgx@param@height>\z@\relax%
1708      \svg@tempa{height=\svgx@param@height}%
1709      \fi%
1710      \ifdim\dimexpr\svgx@param@scale\p@\relax=\p@\relax\else%
1711      \svg@tempa{scale=\svgx@param@scale}%
1712      \fi%
1713      \def\svg@tempb{\svg@param@pretex}%
1714      \ifx\svgx@param@pretex\svg@tempb\relax%
1715      \let\svgx@param@pretex\svg@param@pretex%
1716      \fi%

```



```

1717 \ifx\svgx@param@pretex\relax\else%
1718 \svg@tempa{pretex=\unexpanded\expandafter{\svgx@param@pretex}}%
1719 \fi%
1720 \def\svg@tempb{\svg@param@apptex}%
1721 \ifx\svgx@param@apptex\svg@tempb\relax%
1722 \let\svgx@param@apptex\svg@param@apptex%
1723 \fi%
1724 \ifx\svgx@param@apptex\relax\else%
1725 \svg@tempa{apptex=\unexpanded\expandafter{\svgx@param@apptex}}%
1726 \fi%

```

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

```

1727 \let\svg@tempa\@empty%
1728 \if@svg@ink@latex%
1729 \ifstr{\svg@ink@format}{pdf}{%
1730 \ifnum\value{svg@param@lastpage}>\z@ \relax%
1731 \edef\svg@tempa{lastpage=\the\value{svg@param@lastpage}}%
1732 \else%
1733 \ifnum\value{svg@param@lastpage}=\z@ \relax%
1734 \def\svg@tempa{lastpage=true}%
1735 \else%
1736 \def\svg@tempa{lastpage=false}%
1737 \fi%
1738 \fi%
1739 }{}%
1740 \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.

```

1741 \ifx\svg@tempa\@empty%
1742 \def\svg@tempa{\string\svgxsetbox{#1}}%
1743 \else%
1744 \edef\svg@tempa{\noexpand\string\noexpand\svgxsetbox[\svg@tempa]{#1}}%
1745 \fi%
1746 \immediate\write\svgx@stream@out{\svg@tempa}%

```

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

```

1747 \if@svg@ink@latex%
1748 \IfFileExists{xr.sty}{%
1749 \immediate\write\svgx@stream@out{%
1750 \string\usepackage{xr}^^J%
1751 \string\externaldocument{\jobname}^^J%
1752 }%
1753 }{}%
1754 \fi%
1755 \immediate\write\svgx@stream@out{%
1756 \string\AtBeginDocument{\@percentchar^^J%
1757 \space\space\svg@tempa\@percentchar^^J%
1758 }^^J^^J%
1759 \string\begin{document}^^J%
1760 \string\pagestyle{empty}^^J%
1761 \string\svgxoutputbox\@percentchar^^J%
1762 \string\end{document}%
1763 }%
1764 \immediate\closeout\svgx@stream@out%

```

```
1765 \endgroup%
```

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

```
1766 \ifstr{\svgx@format\svgx@cnv@format}{-}{%
1767 \PackageWarning{svg-extract}{%
1768 Both keys 'extractformat' and 'convertformat' are\MessageBreak%
1769 empty, so nothing to do so far%
1770 }%
1771 }
```

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**.

```
1772 \svg@quotes@remove{\svgx@out@path}%
1773 \svg@quotes@remove{\svgx@out@name}%
```

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

```
1774 \edef\svg@tempb{%
1775 \noexpand\svg@shell@mkdir{\svgx@out@path}%
1776 }%
1777 \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`.

```
1778 \edef\svg@tempb{%
1779 \noexpand\PackageInfo{svg-extract}{%
1780 Running LaTeX (\svgx@latex@exe) for graphic extraction%
1781 \ifx\svgx@latex@opt\@empty\else%
1782 \MessageBreak with added options '\svgx@latex@opt'%
1783 \fi%
1784 }%
1785 }%
1786 \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
1787 \edef\svg@tempb{%
1788 \noexpand\ShellEscape{%
1789 \svgx@latex@exe\space\svgx@latex@opt\space%
1790 "\svgx@out@name.\svgx@latex@ext"%
1791 }%
1792 }%
1793 \loop\ifnum\value{svgx@runs}>\z@ \relax%
1794 \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
1795 \advance\c@svgx@runs\m@ne%
1796 \repeat%
```

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

```
1797 \def\svg@tempa##1##2##3{%
1798 \edef\svg@tempb{%
1799 \noexpand\ShellEscape{%
1800 \@nameuse{svgx@##1@exe}\space\@nameuse{svgx@##1@opt}\space%
1801 "\svgx@out@name.##2"%
1802 }%
1803 }%
```

```

1804 \AfterReadingMainAux{\PackageInfo{svg-extract}{Running ##1}}%
1805 \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
1806 }%
1807 \@svg@tempswafalse%
1808 \ifxetex\else\ifpdf\else%
1809 \@svg@tempswattrue%
1810 \fi\fi%
1811 \if@svg@tempswa%
1812 \svg@tempa{dvips}{dvi}{ps}%
1813 \svgx@ifinlist{eps}{\svgx@format}{\svg@tempa{pstoeps}{ps}{eps}}{}%
1814 \svgx@ifinlist{pdf}{\svgx@format}{\svg@tempa{pstopdf}{ps}{pdf}}{}%
1815 \else%
1816 \svgx@ifinlist{eps}{\svgx@format}{\svg@tempa{pdftoeps}{pdf}{eps}}{}%
1817 \svgx@ifinlist{ps}{\svgx@format}{\svg@tempa{pdftops}{pdf}{ps}}{}%
1818 \fi%

```

Now the desired conversion tool is invoked if requested.

```

1819 \if@svgx@cnv@run%

```

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

```

1820 \ifx\svgx@cnv@dpi\relax%
1821 \ifx\svgx@cnv@dpi@png\@undefined%
1822 \def\svgx@cnv@dpi@png{300}%
1823 \fi%
1824 \fi%

```

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

```

1825 \expandafter\svgx@cnv@get@informat\expandafter{\svgx@format}%

```

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

```

1826 \@for\svg@tempa:=\svgx@cnv@format\do{%
1827 \ifx\svg@tempa\@empty\else%
1828 \expandafter\svgx@ifinlist\expandafter{\svg@tempa}{\svgx@format}{%
1829 \PackageWarning{svg-extract}{%
1830 File type ‘\svg@tempa’ was specified for option\MessageBreak%
1831 ‘extractformat’ (\svgx@format) as well as for \MessageBreak
1832 option ‘convertformat’ (\svgx@cnv@format) so the\MessageBreak%
1833 conversion won’t be done%
1834 }%
1835 }{%
1836 \edef\svg@tempb{%
1837 \noexpand\PackageInfo{svg-extract}{%
1838 Converting ‘\svgx@out@name.\svgx@cnv@informat’\MessageBreak%
1839 to ‘\svgx@out@name.\svg@tempa’%
1840 }%
1841 }%
1842 \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
1843 \edef\svg@tempb{%
1844 \noexpand\ShellEscape{%
1845 \svgx@cnv@cmd{\svgx@out@name}{\svgx@cnv@informat}{\svg@tempa}%
1846 }%
1847 }%
1848 \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
1849 }%
1850 \fi%

```

```

1851     }%
1852     \fi%

```

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

```

1853     \edef\svg@tempa{\svgx@format\if@svgx@cnv@run,\svgx@cnv@format\fi}%
1854     \@for\svg@tempb:=\svg@tempa\do{%
1855         \ifx\svg@tempb\@empty\else%
1856             \edef\svg@tempb{%
1857                 \noexpand\svgx@move{\svgx@out@name}{\svg@tempb}{\svgx@out@path}%
1858             }%
1859             \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
1860         \fi%
1861     }%

```

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

```

1862     \@for\svg@tempa:=\svgx@clean\do{%
1863         \expandafter\svgx@ifinlist\expandafter{\svg@tempa}{\svg@tempb}{\fi}%
1864         \edef\svg@tempb{%
1865             \noexpand\IfFileExists{"\svgx@out@name".\svg@tempa}{%
1866                 \noexpand\svgx@shell@rm{\svgx@out@name.\svg@tempa}%
1867             }{\fi}%
1868         }%
1869         \expandafter\AtEndDocument\expandafter{%
1870             \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
1871         }%
1872     }%
1873 }%
1874 }%
1875 \fi%
1876 }
1877 </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`.

```

1878 \newcommand*\svgx@out@sec{unknown}
1879 \newcommand*\svgx@get@out@sec{%
1880     \begingroup%
1881     \def\svg@tempa{%
1882         \@for\svg@tempb:={%
1883             part,chapter,section,subsection,subsubsection,paragraph,subparagraph%
1884         }\do{%
1885             \ifx\svg@tempb\@empty\else%
1886                 \scr@ifundefinedorrelax{the\svg@tempb}{\fi}%
1887                 \ifnum\value{\svg@tempb}>\z@&relax%
1888                     \edef\svg@tempa{\svg@tempb}%
1889                 \fi%
1890             }%
1891         \fi%
1892     }%
1893     \edef\svg@tempb{%
1894         \endgroup%
1895         \ifx\svg@tempa\@empty\else%
1896             \def\noexpand\svgx@out@sec{\csname the\svg@tempa\endcsname}%
1897         \fi%
1898     }%

```

```
1899 \svg@tempb%
1900 }
```

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

```
1901 \newif\if@svgx@classfound
1902 \newcommand*\svgx@documentclass{}
1903 \def\svgx@documentclass#1\documentclass#2\documentclass#3\@nil{%
1904 \IfArgIsEmpty{#2}{-}{\@svgx@classfoundtrue}%
1905 }
```

`\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
1906 \newcommand*\svgx@read@preamble@till[2]{%
1907 \svgx@read@preamble@skip#1\@nil{till}{#2}%
1908 }
1909 \newcommand*\svgx@read@preamble@from[2]{%
1910 \svgx@read@preamble@skip#1\@nil{from}{#2}%
1911 }
```

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

```
1912 \newcommand*\svgx@read@preamble@skip{}
1913 \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.

```
1914 \def\svg@tempa##1{%
1915 \def\svg@tempa####1##1####2##1####3\@nil{%
1916 \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`.

```
1917 \ifstr{#2}{till}{%
1918 \IfArgIsEmpty{####1}{-}{%
1919 \immediate\write\svgx@stream@out{####1}%
1920 }%
1921 \@svgx@preamble@writefalse%
1922 }{%
```

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

```
1923 \ifstr{#2}{from}{%
1924 \IfArgIsEmpty{####2}{-}{%
1925 \def\svgx@read@line{%
1926 }{%
1927 \def\svgx@read@line{####2}%
1928 }%
1929 \@svgx@preamble@writetrue%
1930 }{}%
1931 }%
```

Additional stuff which should be done.

```
1932      #3%
1933      }%
1934      }%
1935      }%
```

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

```
1936 \edef\svg@tempb{\expandafter\detokenize\expandafter{#1}}%
1937 \expandafter\svg@tempa\expandafter{\svg@tempb}%
```

Calling the created macro.

```
1938 \edef\svg@tempb{%
1939   \expandafter\detokenize\expandafter{\svgx@read@line}\svg@tempb\svg@tempb%
1940 }%
1941 \expandafter\svg@tempa\svg@tempb\@nil%
1942 }
```

`\svgx@cnv@informat`
`\svgx@cnv@get@informat`

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

```
1943 \newcommand*\svgx@cnv@informat{}
1944 \newcommand*\svgx@cnv@get@informat[1]{%
1945   \begingroup%
1946     \def\svg@tempa##1,##2\@nil{%
1947       \def\svg@tempa{##1}%
1948     }%
1949     \svg@tempa#1,\@nil%
1950     \edef\svg@tempa{%
1951       \noexpand\endgroup%
1952       \noexpand\def\noexpand\svgx@cnv@informat{\svg@tempa}%
1953     }%
1954     \svg@tempa%
```

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

```
1955 \ifx\svgx@cnv@informat\@empty%
1956   \renewcommand*\svgx@cnv@informat{pdf}%
1957   \ifxetex\else\ifpdf\else%
1958     \renewcommand*\svgx@cnv@informat{ps}%
1959   \fi\fi%
1960 \fi%
1961 }
```

`\svgx@magick@cmd`
`\svgx@gs@cmd`

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

```
1962 \ifx\svgx@magick@exe\@empty
1963   \ifwindows
1964     \renewcommand*\svgx@magick@exe{magick}
1965   \else
1966     \renewcommand*\svgx@magick@exe{convert}
1967   \fi
1968 \fi
1969 \newcommand*\svgx@magick@cmd[3]{%
1970   \svgx@magick@exe\space%
```

```

1971 \svgx@useformatkey{svgx@cnv@dpi}{#3}{-density }%
1972 \svgx@useformatkey{svgx@magick@set}{#3}{}%
1973 "#1.#2"\space%
1974 \svgx@useformatkey{svgx@magick@opr}{#3}{}%
1975 "#1.#3"%
1976 }

1977 \ifx\svgx@gs@exe\empty
1978 \ifwindows
1979 \renewcommand*\svgx@gs@exe{gswin64c}
1980 \else
1981 \renewcommand*\svgx@gs@exe{gs}
1982 \fi
1983 \fi
1984 \newcommand*\svgx@gs@cmd[3]{%
1985 \svgx@gs@exe\space-dSAFER -dBATC -dNOPAUSE\space%
1986 \svgx@useformatkey{svgx@gs@device}{#3}{-sDEVICE=}%
1987 \svgx@useformatkey{svgx@cnv@dpi}{#3}{-r}%
1988 \svgx@useformatkey{svgx@gs@opt}{#3}{}%
1989 -sOutputFile="#1.#3"\space"#1.#2"%
1990 }

```

\svgx@move If the file doesn't exist

```

1991 \newcommand*\svgx@move[3]{%
1992 \begingroup%
1993 \IfFileExists{"#1".#2}{%
1994 \svg@shell@move{#1.#2}{#3#1.#2}%
1995 }{%
1996 \edef\svg@tempa{#2}%
1997 \@svg@tempswafalse%
1998 \expandafter\svgx@ifinlist\expandafter{\svg@tempa}{\svgx@cnv@format}{%
1999 \@svg@tempswatrue%
2000 \def\svg@tempb{conversion}%
2001 }{%
2002 \expandafter\svgx@ifinlist\expandafter{\svg@tempa}{pdf,ps,eps}{%
2003 \@svg@tempswatrue%
2004 \def\svg@tempb{extraction}%
2005 }{%
2006 }%
2007 \if@svg@tempswa%
2008 \edef\svg@tempb{%
2009 The graphic file \svg@tempb\space failed\MessageBreak%
2010 for '#1.#2'\MessageBreak%
2011 Troubleshooting: Please check the log file how the\MessageBreak%
2012 invocation of the extraction took place and try\MessageBreak%
2013 to execute it yourself in the terminal%
2014 }%
2015 \else%
2016 \def\svg@tempb{%
2017 The extraction to format '#2' failed\MessageBreak%
2018 for '#1.#2'\MessageBreak%
2019 Only file types 'pdf,ps,eps' are supported for\MessageBreak%
2020 key 'exportformat'%
2021 }%
2022 \fi%
2023 \PackageWarning{svg-extract}{\svg@tempb}%
2024 }%
2025 \endgroup%

```

`\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.

```

2027 \newcommand*\svgx@ifinlist[2]{%
2028   \begingroup%
2029   \def\svg@tempa##1,#1,##2\@nil{%
2030     \IfArgIsEmpty{##2}{%
2031       \aftergroup\@secondoftwo%
2032     }{%
2033       \aftergroup\@firstoftwo%
2034     }%
2035   }%
2036   \expandafter\svg@tempa\expandafter,##2,#1,\@nil%
2037 \endgroup%
2038 }
```

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

```

2039 \newcommand*\svgx@onlywindows[1]{}
2040 \AfterPackage*{ifplatform}{\renewcommand*\svgx@onlywindows[1]{\ifwindows#1\fi}}
```

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

```

2041 \newcommand*\svgx@ifkeyandval[3]{%
2042   \def\svg@tempa##1=##2=##3\@nil{\ifstr{##3}{=}{##2}{##3}}%
2043   \svg@tempa#1==\@nil%
2044 }
```

`\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`.

```

2045 \newcommand*\svgx@cnv@get@dpi[1]{%
2046   \begingroup%
2047   \def\svg@tempa##1dpi##2x##3dpi##4\@nil{%
2048     \edef\svg@tempa{##1}%
2049     \if@svg@tempswa as \iftrue means, a valid value was found.
```

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

```

2049     \@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.

```

2050     \ifnumber{##1}{%
2051       \IfArgIsEmpty{##3}{\@svg@tempswatrue}{%
2052         \ifnumber{##3}{\edef\svg@tempa{##1x##3}}{%
2053           }%
2054         }{}%
2055       \if@svg@tempswa\else%
2056         \expandafter\svg@tempb\svg@tempa xx\@nil%
2057       \fi%
2058     }%
```


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

```

2059 \def\svg@tempb##1x##2x##3\@nil{%
2060 \ifstr{##3}{x}{%
2061 \@svg@tempswattrue%
2062 \IfArgIsEmpty{##1}{\@svg@tempswafalse}%
2063 \ifnumber{##1}{\@svg@tempswafalse}%
2064 }%
2065 \IfArgIsEmpty{##2}{\@svg@tempswafalse}%
2066 \ifnumber{##2}{\@svg@tempswafalse}%
2067 }%
2068 \if@svg@tempswa%
2069 \edef\svg@tempa{##1x##2}%
2070 \fi%
2071 }{}%
2072 }%
2073 \IfArgIsEmpty{#1}{%
2074 \let\svg@tempa\@empty%
2075 }{%
2076 \lowercase{\svg@tempa#1dpi#1xdpi\@nil}%
2077 \if@svg@tempswa\else%
2078 \let\svg@tempa\relax%
2079 \fi%
2080 }%
2081 \edef\svg@tempb{%
2082 \noexpand\endgroup%
2083 \ifx\svg@tempa\relax%
2084 \noexpand\let\noexpand\svg@tempa\noexpand\relax%
2085 \else%
2086 \noexpand\def\noexpand\svg@tempa{\svg@tempa}%
2087 \fi%
2088 }%
2089 \svg@tempb%
2090 }

```

`\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.

```

2091 \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`.

```

2092 \svgx@ifkeyandval{#1}{%
2093 \svg@ifvalueisrelax{##2}{%
2094 \expandafter\let\csname #2@##1\endcsname\relax%
2095 }{%
2096 \@namedef{#2@##1}{##2}%
2097 }%

```

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

```

2098 }{%
2099 \svg@ifvalueisrelax{##1}{%
2100 \expandafter\let\csname #2\endcsname\relax%
2101 }{%
2102 \@namedef{#2}{##1}%

```

```

2103 }%
2104 }%
2105 }

```

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.

```

2106 \newcommand*\svgx@useformatkey[3]{%
2107   \scr@ifundefinedorrelax{#1@#2}{%
2108     \scr@ifundefinedorrelax{#1}{%}%
2109     \expandafter\ifx\csname #1\endcsname \@empty\else%
2110       #3\@nameuse{#1}\space%
2111     \fi%
2112   }%
2113   \scr@ifundefinedorrelax{#1@#2+}{%}%
2114   \expandafter\ifx\csname #1@#2+\endcsname \@empty\else%
2115     #3\@nameuse{#1@#2+}\space%
2116   \fi%
2117 }%
2118 }{%

```

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

```

2119   \expandafter\ifx\csname #1@#2\endcsname \@empty\else%
2120     #3\@nameuse{#1@#2}\space%
2121   \fi%
2122 }%
2123 }

```

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.

`\svgxsetbox` Within the preamble of the auxiliary L^AT_EX file, the desired graphic is used to setup a box, which is used both to define the papersize as well as for the output itself.

```

2124 \newbox\svgx@box
2125 \newcommand*\svgxsetbox[2][ ]{%
2126   \sbox\svgx@box{\svg@@input[{#1},draft=false]{#2}}%

```

Setting crop-mode for the TUD-Script-classes.

```

2127   \csname @tud@x@standalone@cropttrue\endcsname%
2128   \svgxsetpapersize%
2129 }

```

`\svgxsetpapersize` This macro sets all well known length macros for defining the paper size as well as the type area to the size of `\svgx@box`.

```

2130 \newcommand*\svgxsetpapersize{%
2131   \setlength\paperwidth{\the\wd\svgx@box}%
2132   \ifx\stockwidth\@undefined\else%
2133     \setlength\stockwidth{\paperwidth}%
2134   \fi%

```

```

2135 \ifx\mediawidth\@undefined\else%
2136 \setlength\mediawidth{\paperwidth}%
2137 \fi%
2138 \setlength\textwidth{\paperwidth}%
2139 \setlength\paperheight{\the\dimexpr\ht\svgx@box+\dp\svgx@box\relax}%
2140 \ifx\stockheight\@undefined\else%
2141 \setlength\stockheight{\paperheight}%
2142 \fi%
2143 \ifx\mediaheight\@undefined\else%
2144 \setlength\mediaheight{\paperheight}%
2145 \fi%
2146 \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.

```

2147 \hoffset=-1in%
2148 \oddsidemargin=\z@%
2149 \evensidemargin=\z@%
2150 \voffset=-1in%
2151 \topmargin=\z@%
2152 \headheight=\z@%
2153 \headsep=\z@%
2154 \topskip=\z@%
2155 \footskip=\z@%
2156 \marginparsep=\z@%
2157 \marginparwidth=\z@%
2158 \marginparpush=\z@%
2159 }
2160 \@onlypreamble\svgxsetpapersize

```

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

```

2161 \newif\if@svgx@beamer
2162 \@ifclassloaded{beamer}{\@svgx@beamertrue}{}%
2163 \newcommand*\svgxoutputbox{%
2164 \begingroup%
2165 \setlength\parindent{\z@}%
2166 \setlength\parskip{\z@}%
2167 \setlength\parfillskip{\z@}%
2168 \if@svgx@beamer%
2169 \setbeamertemplate{navigationsymbols}{}%
2170 \begin{frame}[plain]%
2171 \usebox\svgx@box%
2172 \end{frame}%
2173 \else%
2174 \usebox\svgx@box%
2175 \fi%
2176 \endgraf%
2177 \endgroup%
2178 }

```

D. Processing Options

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

```

2179 (*base)

```

```

2180 \FamilyExecuteOptions{SVG}{%
2181   inkscape=true,inkscapepath=basesubdir,inkscapelatex=true,%
2182   inkscapearea=drawing,usecolor=true,usetransparent=true%
2183 }
2184 \end{base}
2185 \end{extract}
2186 \FamilyExecuteOptions{SVG}{%
2187   extract=true,extractpath=basesubdir,extractruns=2,extractname=namenumbered,%
2188   convert=magick,convert=false,%
2189   gsdevice={png=png16m},gsdevice={jpeg=jpeg},gsdevice={jpg=jpeg},%
2190   gsdevice={tif=tiff48nc},gsdevice={tiff=tiff48nc},%
2191   gsdevice={eps=eps2write},gsdevice={ps=ps2write}%
2192 }
2193 \end{extract}
2194 \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.

```

2195 \ifnum\pdf@shellescape=\@ne\relax\else%
2196   \expandafter\endinput%
2197 \fi
2198 \RequirePackage{ifplatform}[2010/10/22]

```

```

\svg@shell@mkdir The platform dependent commands for file access.
\svg@shell@@mkdir
  \svg@shell@mv
\svg@shell@@mv
  \svg@shell@rm
\svg@shell@@rm
2199 \ifwindows
2200   \newcommand*\svg@shell@@mkdir[1]{if not exist "#1" mkdir "#1"}
\svg@shell@@mv
2201   \newcommand*\svg@shell@@mv{move}
\svg@shell@@rm
2202   \newcommand*\svg@shell@@rm{del}
\svg@shell@@rm
2203 \else
2204   \newcommand*\svg@shell@@mkdir[1]{mkdir -p "#1"}
2205   \newcommand*\svg@shell@@mv{mv}
2206   \newcommand*\svg@shell@@rm{rm}
2207 \fi

```

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

```

2208 \newcommand*\svg@shell@@mkdir[1]{%
2209   \begin{group}%
2210     \edef\svg@tempa{#1}%
2211     \svg@quotes@remove{\svg@tempa}%
2212     \@svg@tempswatrue%
2213     \ifstr{\svg@tempa}{.}{\@svg@tempswafalse}{%
2214       \ifstr{\svg@tempa}{./}{\@svg@tempswafalse}{%
2215         }%
2216       \if@svg@tempswa%
2217         \ShellEscape{\svg@shell@@mkdir{\svg@tempa}}%
2218       \fi%
2219     \end{group}%
2220 }

```

Commands for moving and deleting files.

```

2221 \newcommand*\svg@shell@move[2]{%
2222   \ShellEscape{\svg@shell@mv\space"#1"\space"#2"}%
2223 }
2224 \newcommand*\svg@shell@rm[1]{%
2225   \ShellEscape{\svg@shell@rm\space"#1"}%
2226 }

```

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		I	
apptex (opt.)	6, 277	\ifsvg@draft	314
		\ifsvg@file@found	382
		\ifsvg@ink@run	688
		\ifsvg@quotes@found	335
		\ifsvg@tempwa	25
		\ifsvg@use@transparent	31
		\ifsvg@use@xcolor	31
		\ifsvgx@beamer	2161
		\ifsvgx@classfound	1901
		\ifsvgx@cnv@run	1156
		\ifsvgx@out@sec	1328
		\ifsvgx@preamble@write	1556
		\ifsvgx@run	952
		\includeinkscape	7–8 , 622
		angle (param.)	7 , 646
		apptex (param.)	7 , 646
		clean (param.)	8 , 646
		convert (param.)	8 , 646
		convertdpi (param.)	8 , 646
		convertformat (param.)	8 , 646
		draft (param.)	7 , 646
		exclude (param.)	8 , 646
		extract (param.)	8 , 646
		extractapptex (param.)	8 , 646
		extractformat (param.)	8 , 646
		extractheight (param.)	8 , 646
		extractpreamble (param.)	8 , 646
		extractpretex (param.)	8 , 646
		extractruns (param.)	8 , 646
		extractscale (param.)	8 , 646
		extractwidth (param.)	8 , 646
		gsdevice (param.)	8 , 646
		gsopt (param.)	8 , 646
		height (param.)	7 , 646
		inkscapeformat (param.)	7 , 646
		inkscapelatex (param.)	7 , 646
		lastpage (param.)	7 , 646
		latexopt (param.)	8 , 646
		magickoperator (param.)	8 , 646
		magicksetting (param.)	8 , 646
		origin (param.)	7 , 646
		pretex (param.)	7 , 646
C			
clean (opt.)	10 , 1493		
clear (opt.)	1493		
convert (opt.)	10 , 1156		
convertdensity (opt.)	1259		
convertdpi (opt.)	11 , 1259		
convertformat (opt.)	10 , 1232		
counters:			
svg@param@lastpage	300		
svgx@out@count	1328		
svgx@runs	1052		
D			
draft (opt.)	6 , 314		
dvipsopt (opt.)	10 , 1107		
E			
end (opt.)	1025		
eps (opt.)	984		
exclude (opt.)	10 , 1514		
extract (opt.)	8 , 952		
extractapptex (opt.)	9 , 1457		
extractformat (opt.)	9 , 984		
extractheight (opt.)	9 , 1394		
extractname (opt.)	8 , 1328		
extractpath (opt.)	8 , 1328		
extractpostex (opt.)	1457		
extractpreamble (opt.)	9 , 1025		
extractpreambleend (opt.)	9 , 1025		
extractpretex (opt.)	9 , 1457		
extractruns (opt.)	9 , 1052		
extractscale (opt.)	9 , 1394		
extractwidth (opt.)	9 , 1394		
G			
gsdevice (opt.)	12 , 1306		
gsexex (opt.)	12 , 1306		
gsopt (opt.)	12 , 1306		
H			
height (opt.)	6 , 236		

scale (param.)	7, 646		
width (param.)	7, 646		
\includesvg	6, 8, 573		
angle (param.)	7, 608		
apptex (param.)	6, 576		
clean (param.)	8, 576		
convert (param.)	8, 576		
convertdpi (param.)	8, 576		
convertformat (param.)	8, 576		
draft (param.)	6, 576		
exclude (param.)	8, 576		
extract (param.)	8, 576		
extractapptex (param.)	8, 576		
extractformat (param.)	8, 576		
extractheight (param.)	8, 576		
extractpreamble (param.)	8, 576		
extractpretex (param.)	8, 576		
extractruns (param.)	8, 576		
extractscale (param.)	8, 576		
extractwidth (param.)	8, 576		
gsdevice (param.)	8, 576		
gsopt (param.)	8, 576		
height (param.)	6, 576		
inkscape (param.)	6, 576		
inkscapearea (param.)	6, 576		
inkscapedpi (param.)	6, 576		
inkscapeformat (param.)	6, 576		
inkscapelatex (param.)	6, 576		
inkscapeopt (param.)	6, 576		
lastpage (param.)	7, 605		
latexopt (param.)	8, 576		
magickoperator (param.)	8, 576		
magicksetting (param.)	8, 576		
origin (param.)	7, 608		
pretex (param.)	6, 576		
scale (param.)	6, 576		
width (param.)	6, 576		
inkscape (opt.)	4, 55		
inkscapearea (opt.)	5, 153		
inkscapedensity (opt.)	167		
inkscapedpi (opt.)	5, 167		
inkscapeexe (opt.)	5, 184		
inkscapeformat (opt.)	5, 130		
inkscapelatex (opt.)	5, 151		
inkscapename (opt.)	206		
inkscapeopt (opt.)	5, 184		
inkscapepath (opt.)	5, 206		
L			
lastpage (opt.)	6, 300		
latexexe (opt.)	10, 1069		
latexext (opt.)	10, 1069		
latexopt (opt.)	10, 1069		
M			
magickexe (opt.)	11, 1284		
magickoperator (opt.)	11, 1284		
magicksetting (opt.)	11, 1284		
N			
name (opt.)	1328		
nottransparent (opt.)	3, 31		
noxcolor (opt.)	3, 31		
O			
off (opt.)	8, 128 , 980		
on (opt.)	8, 128 , 980		
options:			
apptex	6, 277		
clean	10, 1493		
clear	1493		
convert	10, 1156		
convertdensity	1259		
convertdpi	11, 1259		
convertformat	10, 1232		
draft	6, 314		
dvipsopt	10, 1107		
end	1025		
eps	984		
exclude	10, 1514		
extract	8, 952		
extractapptex	9, 1457		
extractformat	9, 984		
extractheight	9, 1394		
extractname	8, 1328		
extractpath	8, 1328		
extractpostex	1457		
extractpreamble	9, 1025		
extractpreambleend	9, 1025		
extractpretex	9, 1457		
extractruns	9, 1052		
extractscale	9, 1394		
extractwidth	9, 1394		
gsdevice	12, 1306		
gsexex	12, 1306		
gsopt	12, 1306		
height	6, 236		
inkscape	4, 55		
inkscapearea	5, 153		
inkscapedensity	167		
inkscapedpi	5, 167		
inkscapeexe	5, 184		
inkscapeformat	5, 130		
inkscapelatex	5, 151		
inkscapename	206		
inkscapeopt	5, 184		
inkscapepath	5, 206		
lastpage	6, 300		
latexexe	10, 1069		
latexext	10, 1069		
latexopt	10, 1069		
magickexe	11, 1284		
magickoperator	11, 1284		
magicksetting	11, 1284		
name	1328		
nottransparent	3, 31		
noxcolor	3, 31		
off	8, 128 , 980		

on	8, 128 , 980
path	1328
pdf	984
pdflatex	1069
pdftoeptopt	10 , 1107
pdftops	1107
pdftopsopt	10 , 1107
png	1232
postex	277
preamble	1025
pretex	6 , 277
pstoepsot	10 , 1107
pstopdfot	10 , 1107
scale	6 , 236
svgpath	194
usetransparent	3 , 31
usexcolor	3 , 31
width	6 , 236

P

parameters:

angle- \includeinkscape	7 , 646
angle- \includesvg	7 , 608
apptex- \includeinkscape	7 , 646
apptex- \includesvg	6 , 576
clean- \includeinkscape	8 , 646
clean- \includesvg	8 , 576
convert- \includeinkscape	8 , 646
convert- \includesvg	8 , 576
convertdpi- \includeinkscape	8 , 646
convertdpi- \includesvg	8 , 576
convertformat- \includeinkscape	8 , 646
convertformat- \includesvg	8 , 576
draft- \includeinkscape	7 , 646
draft- \includesvg	6 , 576
exclude- \includeinkscape	8 , 646
exclude- \includesvg	8 , 576
extract- \includeinkscape	8 , 646
extract- \includesvg	8 , 576
extractapptex- \includeinkscape	8 , 646
extractapptex- \includesvg	8 , 576
extractformat- \includeinkscape	8 , 646
extractformat- \includesvg	8 , 576
extractheight- \includeinkscape	8 , 646
extractheight- \includesvg	8 , 576
extractpreamble- \includeinkscape	8 , 646
extractpreamble- \includesvg	8 , 576
extractpretex- \includeinkscape	8 , 646
extractpretex- \includesvg	8 , 576
extractruns- \includeinkscape	8 , 646
extractruns- \includesvg	8 , 576
extractscale- \includeinkscape	8 , 646
extractscale- \includesvg	8 , 576
extractwidth- \includeinkscape	8 , 646
extractwidth- \includesvg	8 , 576
gsdevice- \includeinkscape	8 , 646
gsdevice- \includesvg	8 , 576
gsot- \includeinkscape	8 , 646

gsot- \includesvg	8 , 576
height- \includeinkscape	7 , 646
height- \includesvg	6 , 576
inkscape- \includesvg	6 , 576
inkscapearea- \includesvg	6 , 576
inkscapedpi- \includesvg	6 , 576
inkscapeformat- \includeinkscape	7 , 646
inkscapeformat- \includesvg	6 , 576
inkscapelatex- \includeinkscape	7 , 646
inkscapelatex- \includesvg	6 , 576
inkscapeopt- \includesvg	6 , 576
lastpage- \includeinkscape	7 , 646
lastpage- \includesvg	7 , 605
latexopt- \includeinkscape	8 , 646
latexopt- \includesvg	8 , 576
magickoperator- \includeinkscape	8 , 646
magickoperator- \includesvg	8 , 576
magicksetting- \includeinkscape	8 , 646
magicksetting- \includesvg	8 , 576
origin- \includeinkscape	7 , 646
origin- \includesvg	7 , 608
pretex- \includeinkscape	7 , 646
pretex- \includesvg	6 , 576
scale- \includeinkscape	7 , 646
scale- \includesvg	6 , 576
width- \includeinkscape	7 , 646
width- \includesvg	6 , 576
path (opt.)	1328
pdf (opt.)	984
pdflatex (opt.)	1069
pdftoeptopt (opt.)	10 , 1107
pdftops (opt.)	1107
pdftopsopt (opt.)	10 , 1107
png (opt.)	1232
postex (opt.)	277
preamble (opt.)	1025
pretex (opt.)	6 , 277
pstoepsot (opt.)	10 , 1107
pstopdfot (opt.)	10 , 1107

S

scale (opt.)	6 , 236
\setsvg	557
\svg@deprecated@key	15
\svg@deprecated@param	666
\svg@dummy@key	927
\svg@extract	1556
\svg@file@base	382
\svg@file@missing	485
\svg@file@name	382
\svg@file@path	382
\svg@file@suffix	382
\svg@filename@parse	444
\svg@get@lastpage	775
\svg@get@path	382
\svg@get@@path	382
\svg@ifilenewer	530
\svg@ifvalueisrelax	371
\svg@includegraphics@file	908

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 [1493](#)
convert (opt.): changed/extended . . . [1156](#)
convertdpi (opt.): new [1259](#)
convertformat (opt.): new [1232](#)
draft (opt.): new [314](#)
dvipsopt (opt.): new [1107](#)
end (opt.): deprecated [1025](#)
eps (opt.): deprecated [984](#)
extract (opt.): new [952](#)
extractapptex (opt.): new [1457](#)
extractformat (opt.): new [984](#)
extractheight (opt.): new [1394](#)
extractname (opt.): new [1328](#)
extractpath (opt.): new [1328](#)
extractpreamble (opt.): new [1025](#)
extractpreambleend (opt.): new [1025](#)
extractpretex (opt.): new [1457](#)
extractruns (opt.): new [1052](#)
extractscale (opt.): new [1394](#)
extractwidth (opt.): new [1394](#)
gsdevice (opt.): new [1306](#)
gsex (opt.): new [1306](#)
gs (opt.): new [1306](#)
height (opt.): new [236](#)
\includeinkscape: new [622](#)
\includesvg:
 changes, especially to optional
 parameters [573](#)
angle (param.): new [608](#)
draft (param.): new [576](#)
height (param.): new [576](#)
inkscape (param.): new [576](#)
inkscapearea (param.): new [576](#)

inksapedpi (param.): new [576](#)
inkscapeformat (param.): new [576](#)
inksapelatex (param.): new [576](#)
inkscapeopt (param.): new [576](#)
lastpage (param.): new [605](#)
origin (param.): new [608](#)
scale (param.): new [576](#)
inkscape (opt.): changed/extended [55](#)
inkscapearea (opt.): new [153](#)
inksapedpi (opt.): new [167](#)
inkscapeexe (opt.): new [184](#)
inkscapeformat (opt.): new [130](#)
inksapelatex (opt.): new [151](#)
inksapename (opt.): new [206](#)
inkscapeopt (opt.): new [184](#)
inkscapepath (opt.): new [206](#)
lastpage (opt.): new [300](#)
latexexe (opt.): new [1069](#)
latexext (opt.): new [1069](#)
latexopt (opt.): new [1069](#)
magickexe (opt.): new [1284](#)
magickoperator (opt.): new [1284](#)
magicksetting (opt.): new [1284](#)
name (opt.):
 deprecated [1328](#)
 support of **subfig** removed [1328](#)
notransparent (opt.): new [31](#)
noxcolor (opt.): new [31](#)
off (opt.): new [128](#), [980](#)
on (opt.): new [128](#), [980](#)
path (opt.): deprecated [1328](#)
pdf (opt.): deprecated [984](#)
pdflatex (opt.): deprecated [1069](#)
pdftoeptsopt (opt.): new [1107](#)
pdftops (opt.): deprecated [1107](#)
pdftopsopt (opt.): new [1107](#)
png (opt.): deprecated [1232](#)
postex (opt.): deprecated [277](#)
preamble (opt.): deprecated [1025](#)
pstoeptsopt (opt.): new [1107](#)
pstopdfopt (opt.): new [1107](#)
scale (opt.): new [236](#)
\setsvg: deprecated [557](#)
\svghidepreambleend: new [1536](#)
\svghidepreamblestart: new [1536](#)
\svgpath: new [559](#)
svgpath (opt.): deprecated [194](#)
\svgsetup: new [557](#)
usetransparent (opt.): new [31](#)
usexcolor (opt.): new [31](#)