

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

Philip Ilten (2012–2016)

Falk Hanisch (2017–)

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

[hanisch.latex@outlook.com](mailto:hanisch.latex@outlook.com)

v2.02e (2020/01/13)

The **svg** package is intended for the automated integration of SVG graphics into  $\text{\LaTeX}$  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  $\text{\LaTeX}$ . 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,  $\text{\LaTeX}$  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

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

A.2.	Helper macros	16
<b>B.</b>	<b>Including SVG files with package <i>svg</i></b>	<b>16</b>
B.1.	Options	16
B.1.1.	The invocation of <i>Inkscape</i>	17
B.1.2.	Setting input folder and file	20
B.1.3.	Setting output folder	21
B.1.4.	Options for the inclusion of graphics	21
B.2.	Handling path information	23
B.3.	Optional Parameters for user commands	31
B.4.	User commands	31
B.5.	Auxiliary macros	34
B.6.	Patches	39
<b>C.</b>	<b>Extracting independent graphic files with <i>svg-extract</i></b>	<b>41</b>
C.1.	Options	41
C.1.1.	Controlling the extract process	41
C.1.2.	Invoking external application for graphic conversion	45
C.1.3.	Setting output folder	48
C.1.4.	Options for the extraction of graphics	49
C.1.5.	Miscellaneous options	52
C.2.	User commands	52
C.3.	Auxiliary macros	53
C.4.	Commands for the separate auxiliary <i>L<sup>A</sup>T<sub>E</sub>X</i> -file	66
<b>D.</b>	<b>Processing Options</b>	<b>67</b>
<b>E.</b>	<b>Macros for file access</b>	<b>68</b>
	<b>Index</b>	<b>69</b>
	<b>Change History</b>	<b>72</b>

# 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<sup>A</sup>T<sub>E</sub>X*, and an image component that can be imported as a PDF file. For further information see the documentation of *svg-inkscape* on CTAN<sup>1</sup>. The procedure described therein is taken up and consistently expanded. Thus, it is now possible to include a SVG file into a *L<sup>A</sup>T<sub>E</sub>X* document where the text within the SVG graphic will be rendered natively by *L<sup>A</sup>T<sub>E</sub>X*.

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*.

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

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

**iftex** for flow control depending on the used L<sup>A</sup>T<sub>E</sub>X engine

**scrbase** for the definition and handling of options in key-value-syntax

**pdftexcmds**, **shellesc** to allocate the same primitives independent of the used L<sup>A</sup>T<sub>E</sub>X engine

**ifplatform** to control the file access depending on the operating system

**trimspaces** to remove unwanted spaces in file paths

**graphicx** for including the graphic files after the **Inkscape** export

**xcolor**, **transparent** are possibly needed by the separate L<sup>A</sup>T<sub>E</sub>X 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<sup>A</sup>T<sub>E</sub>X 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  
`nottransparent` (opt.) (`true/on/yes/false/off/no`) can be assigned to `usexcolor` and `usetransparent`, while `noxcolor` and `nottransparent` 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 `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 L<sup>A</sup>T<sub>E</sub>X document can be reduced to the necessary minimum.

`forced/force/overwrite`

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

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

`pdf/eps/ps/png`

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

`latex/nolatem`

see `inkscapelatem=true/false`

`drawing/page`

see `inkscapearea=drawing/page`

`<integer>dpi`

see `inkscapedpi=<integer>`

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

`svgdir/svgpath`

The PDF/EPS/PS/PNG graphic files as well as the L<sup>A</sup>T<sub>E</sub>X 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.

- inkscape**name (opt.) The file names of the **Inkscape** export are derived from the name of the base SVG file and can be modified with **inkscape**name=*<filename>*. It's possible to use counters for specifying the name of the exported file. Repeatedly specifying the same file name will overwrite previously created files.
- inkscape**exe (opt.) For including a SVG file, **Inkscape** is used to separate the text and image from the SVG file itself. In order to execute the command line tool from shell, the path where the executable is located has to be known to the operating system. You can check this by typing **inkscape -V** into the shell. If this check fails and you don't want to change environment variable **path** on your OS, you can use option **inkscape**exe to set the absolute path where the executable of **Inkscape** is located. The option is set to **inkscape**exe=**inkscape** by default.
- inkscape**format (opt.) With this option, the **Inkscape** export format can be controlled. Valid values are **pdf**, **eps**, **ps** and **png**, where a  $\text{\LaTeX}$  export is not possible for **png** and option **inkscape**latex won't have any effect. By default, **inkscape**format=**pdf** is set unless DVI output was detected. In this case **inkscape**format=**eps** is the default setting.
- inkscape**latex (opt.) If option **inkscape**latex=**true** is set, the output is split into a separate PDF/EPS/PS file (see option **inkscape**format) and a corresponding  $\text{\LaTeX}$  file. This is the default setting. Setting **inkscape**latex=**false** will result in a single PDF/EPS/PS file, where any contained text won't be rendered by  $\text{\LaTeX}$ .
- inkscape**area (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.
- inkscape**dpi (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 **inkscape**dpi=**\relax**.
- inkscape**opt (opt.) You can use this option to pass additional switches to the **Inkscape** command line tool. For further information see the documentation of **Inkscape**<sup>2</sup>.
- svg**extension (opt.) The package assumes SVG files with **.svg** extension as source for the **Inkscape** export. This option can be used to change this behaviour. For example, in order to process **.dia** files instead of **.svg** you could use

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

## 2.3. Options for the graphic inclusion

- width** (opt.) The width of the included graphic file can be specified via the **width** option and the height
- height** (opt.) by the **height** option. If both the width and height are specified, the figure will be scaled
- distort** (opt.) such that neither of the specified dimensions is exceeded, unless option **distort**=**true** is
- scale** (opt.) given.<sup>3</sup> If **width** and/or **height** once have been set, this can be undone by setting them to **Opt** or **\relax**. If neither **width** nor **height** are set, the included graphic file can also be scaled by setting **scale** to a positive real number.
- pretex** (opt.) Commands prior and post to the inclusion of the graphic file may be desired, such as font or
- apptex** (opt.) color commands. The options **pretex** and **apptex** are provided where the  $\text{\LaTeX}$  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:

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

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

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

- draft** (opt.) This option can be used with boolean values and is equal to the identically named option of the **graphicx** package. If the **draft** option is given to **graphicx**, it's activated for **svg** as well.
- lastpage** (opt.) A [bug](https://bugs.launchpad.net/ubuntu/+source/inkscape/+bug/1417470)<sup>4</sup> concerning the L<sup>A</sup>T<sub>E</sub>X export has been reported for **Inkscape** 0.91. It may happen that within the exported L<sup>A</sup>T<sub>E</sub>X file, it's attempted to include more pages of the PDF graphics than actually exist. The **svg** package attempts to bypass the resulting error.

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

## 2.4. Including SVG files

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

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

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

## 2.5. Including already exported SVG files

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

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

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

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



### 3. Usage of package `svg-extract`

This package allows the extraction of independent graphic files out of SVG files which have been included and rendered with  $\text{\LaTeX}$  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  $\text{\LaTeX}$  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  $\text{\LaTeX}$  2 $\epsilon$  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  $\text{\LaTeX}$  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  $\text{\LaTeX}$  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

<code>on</code> (opt.)	This options have to be given while loading the <code>svg-extract</code> package and are intended to toggle the functionality of this package. As both extracting and converting independent graphic files is invoked with every $\text{\LaTeX}$ run when <code>--shell-escape</code> is activated, the option <code>off</code> 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 <code>on</code> can be used to reactivate functionality of this package. This can also be done by using <code>extract=true/false</code> .
<code>off</code> (opt.)	
<code>\svgsetup</code> <code>\includesvg</code> <code>\includeinkscape</code>	With package <code>svg-extract</code> the applicable options for <code>\svgsetup{&lt;options&gt;}</code> as well as parameters for the already described macros <code>\includesvg[&lt;parameters&gt;]{&lt;filename&gt;}</code> and <code>\includeinkscape[&lt;parameters&gt;]{&lt;filename&gt;}</code> are extended. They can be used to control the process of graphic extraction and converting.
<code>extractangle</code> (param.)	With this parameter the graphic is rotated during the extraction process. The value is not inherited from <code>angle</code> if it was given by default. this can be achieved by setting:

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

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

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

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

## 3.2. Extract independent graphic files

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



known within this file. Consequently, the preamble of the current L<sup>A</sup>T<sub>E</sub>X document is used for the extraction of the SVG file by default.

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

`\svghidepreamblestart` In case, the preamble of the current L<sup>A</sup>T<sub>E</sub>X document is used, there are maybe packages included or some parts within the preamble, which should not be used within the separate auxiliary L<sup>A</sup>T<sub>E</sub>X file. These parts can be excluded if they are enclosed by `\svghidepreamblestart` and `\svghidepreambleend`.

For example, your current L<sup>A</sup>T<sub>E</sub>X document uses package **showframe** which causes some problems with the extraction of independent graphic files. So you want to get rid of it within the auxiliary L<sup>A</sup>T<sub>E</sub>X file. This can be done with:

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

**extractruns** (opt.) When extracting independent graphic files by compiling the generated auxiliary L<sup>A</sup>T<sub>E</sub>X file, it's maybe necessary to do multiple L<sup>A</sup>T<sub>E</sub>X 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<sup>A</sup>T<sub>E</sub>X program is used which is set by the

**latexopt** (opt.) **latexexe** option. Depending on the L<sup>A</sup>T<sub>E</sub>X processor used for the current L<sup>A</sup>T<sub>E</sub>X document, it is set to either **pdflatex**, **lualatex**, **xelatex** or **latex** by default. It's also possible to specify additional flags or switches for the L<sup>A</sup>T<sub>E</sub>X runs, which are performed during the extraction process by the **latexopt** option. If you are used to utilize a other file extension for L<sup>A</sup>T<sub>E</sub>X files than `.tex`, option **latexext** can be used like **latexext=ltx**.

**dvipsopt** (opt.) Depending on the used L<sup>A</sup>T<sub>E</sub>X processor, the file type of the extracted graphic differs. In

**pstoeps** (opt.) order to create all formats, requested with option **extractformat**, several converting tools provided by most of the L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> distributions are maybe invoked. These are **dvips**, **ps2eps**, **pstopdf** (opt.) **ps2pdf** and/or **pdftops** and can't be changed. It's only possible to specify additional **pdftops** (opt.) switches for every single tool with **dvipsopt**, **pstoeps** (opt.) **pstopdf** (opt.) **pdftops** (opt.) and **pdftops** (opt.) **pdftops**.

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

**exclude** (opt.) Sometimes it may be necessary to extract and/or convert a SVG file without including it. If the flag **exclude** is specified, the SVG file will not be rendered in the current L<sup>A</sup>T<sub>E</sub>X 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  $\langle option \rangle = \langle value \rangle$  or  $\langle option \rangle = \{ \langle outputformat \rangle = \langle value \rangle \}$  and even  $\langle option \rangle = \{ \langle outputformat \rangle += \langle value \rangle \}$  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 ( $\langle option \rangle = \text{\relax}$ ) or a specific one ( $\langle option \rangle = \{ \langle outputformat \rangle [+]=\text{\relax} \}$ ).

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

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

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

If a setting for a specific output format is given, any 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](#)<sup>5</sup>.

### 3.3.2. Settings for the invocation of *Ghostscript*

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

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

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

## 4. Example

As an minimal example<sup>7</sup> take the following lines of code:

```
\documentclass{article}  
\usepackage[T1]{fontenc}  
\usepackage{svg}  
\usepackage[off]{svg-extract}  
\svgsetup{clean=true}  
%\pdfsuppresswarningpagegroup=1  
\usepackage{relsize}  
\usepackage{subcaption}  
\begin{document}  
\begin{figure}  
  \begin{minipage}{.5\linewidth}  
    \includesvg[width=\linewidth]{svg-example}%  
    \subcaption{This text is too large!}  
  \end{minipage}%  
  \begin{minipage}{.5\linewidth}  
    \includesvg[width=\linewidth,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}
```

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

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

<sup>7</sup>The image used here is a slightly modified version of the image used in the initial documentation on how to include a SVG file in L<sup>A</sup>T<sub>E</sub>X by Johan B. C. Engelen available as package *svg-inkscape* on CTAN.

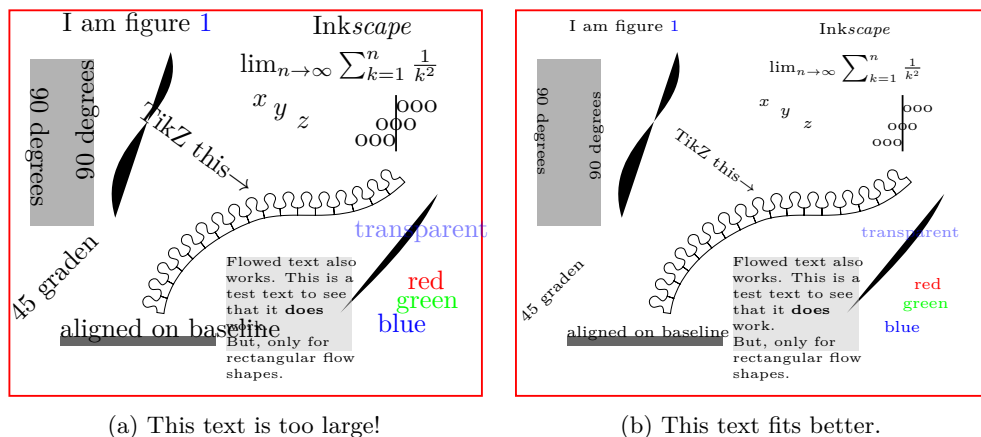


Figure 1: An example figure with L<sup>A</sup>T<sub>E</sub>X support

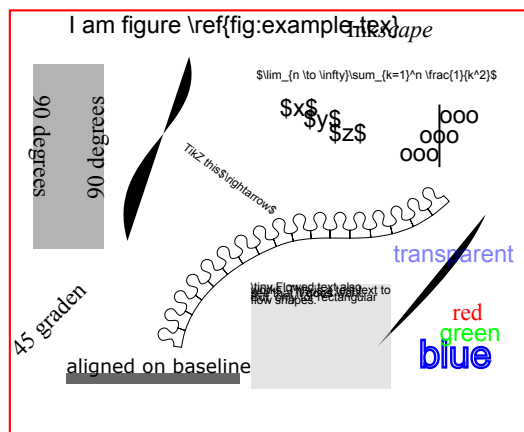


Figure 2: The same example figure without L<sup>A</sup>T<sub>E</sub>X support

```
\end{document}
```

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

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

As you can see, Figure 1a is created with default settings, except for the width specification. So the **Inkscape** export with L<sup>A</sup>T<sub>E</sub>X support is done as well as the extraction of a independent graphic file in PDF format as the **svg-extract** package was loaded.

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

In Figure 2 the same SVG file was used but without the export of a separate L<sup>A</sup>T<sub>E</sub>X 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 pdf $\LaTeX$ 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 pdf $\TeX$  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](https://tex.stackexchange.com/questions/76273/multiple-pdfs-with-page-group-included-in-a-single-page)<sup>8</sup> 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  $\LaTeX$  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  $\LaTeX$  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**<sup>9</sup>. Remember that the font is set by using the function `(TAttFill*)->SetTextFont(i)` with

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

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

---

<sup>8</sup>[http://tex.stackexchange.com/questions/76273/](https://tex.stackexchange.com/questions/76273/multiple-pdfs-with-page-group-included-in-a-single-page)

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

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

Now, you can just use the well-known standard  $\text{\LaTeX}$  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.

The plot was then included within this document using the following  $\text{\LaTeX}$  code

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



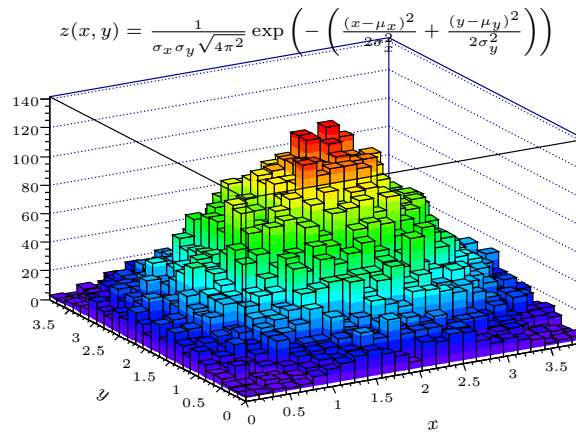


Figure 3: Rendering of a **ROOT** plot—no more *Comic CERNs*

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

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

# Part II.

## Implementation

### A. Initialization

#### A.1. Packages

The package **svg** requires package **iftex** for detecting the used L<sup>A</sup>T<sub>E</sub>X engine, **scrbase** for options processing, **pdftexcmds** for pdfT<sub>E</sub>X primitives when using LuaT<sub>E</sub>X, **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 off with the corresponding options. Package **svg-extract** only needs package **svg** itself.

```
1 < *base>
2 \RequirePackage{iftex}[2019/11/07]
3 \RequirePackage{scrbase}[2019/12/23]
4 \RequirePackage{pdftexcmds}[2016/05/21]
5 \RequirePackage{shellesc}[2016/06/07]
6 \RequirePackage{trimspaces}[2009/09/17]
7 \RequirePackage{graphicx}[1999/02/16]
8 < /base>
9 < *extract>
10 \RequirePackage{svg}[2017/03/27]
11 < /extract>
```

#### A.2. Helper macros

<pre>\svg@tempa \svg@tempb \svg@box \if@svg@tempswa</pre>	<p>Internal temporary macros. The catcode for double quotes are also temporarily changed.</p> <pre>12 \newcommand*\svg@tempa{} 13 \newcommand*\svg@tempb{} 14 \newbox\svg@box 15 \newif\if@svg@tempswa 16 \edef\svg@catcodecodes@restore{% 17   \catcode'\noexpand\"the\catcode'\relax% 18 } 19 \@makeother\"%</pre>
---	--

### B. Including SVG files with package svg

#### B.1. Options

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

```
20 \DefineFamily{SVG}
21 \DefineFamilyMember{SVG}
```

<pre>\svg@deprecated@key</pre>	<p>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.</p>
--------------------------------	--

```
22 \newcommand*\svg@deprecated@key[3][svg]{%
23   \PackageWarning{#1}{%
24     The option key '#2' is deprecated.\MessageBreak%
```

```

25     It's recommended to use '#3'\MessageBreak%
26     instead%
27 }%
28 \FamilyOptions{SVG}{#3}%
29 }

```

Within the exported L<sup>A</sup>T<sub>E</sub>X files of *Inkscape*, some commands are used out of additional packages. But maybe the user doesn't want to load this packages anyways.

<pre> usexcolor (opt.) noxcolor (opt.) \if@svg@use@xcolor usetransparent (opt.) nottransparent (opt.) \if@svg@use@transparent </pre>	<p>Options for preventing packages <b>xcolor</b> and <b>transparent</b> to be loaded.</p> <pre> 30 \newif\if@svg@use@xcolor 31 \FamilyBoolKey{SVG}{usexcolor}{@svg@use@xcolor} 32 \DeclareOption{noxcolor}{\FamilyOptions{SVG}{usexcolor=false}} 33 \newif\if@svg@use@transparent 34 \FamilyBoolKey{SVG}{usetransparent}{@svg@use@transparent} 35 \DeclareOption{nottransparent}{\FamilyOptions{SVG}{usetransparent=false}} </pre>
--	--

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

```

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

```

### 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<sup>A</sup>T<sub>E</sub>X can optionally be used.

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

```

54 \newcommand*\svg@ink@mode{}
55 \DefineFamilyKey{SVG}{inkscape}[true]{%
56   \lowercase{\svg@sanitize@dq\svg@tempb{#1}}%
57   \FamilySetNumerical{SVG}{inkscape}{svg@tempa}{%
58     {false}{0},{off}{0},{no}{0},%
59     {true}{1},{on}{1},{yes}{1},{onlynewer}{1},{newer}{1},%
60     {force}{2},{forced}{2},{overwrite}{2},%
61     {pdf}{3},{eps}{4},{ps}{5},{png}{6},%
62     {drawing}{7},{crop}{7},%
63     {page}{8},{nocrop}{8},%
64     {tex}{9},{latex}{9},{exportlatex}{9},{latexexport}{9},%
65     {notex}{10},{nolatemex}{10},{noexportlatex}{10},{nolatemexexport}{10},%

```

```

66     {\latexnoexport}{10},{raw}{10},{plain}{10},{simple}{10}%
67   }\svg@tempb}%
68   \ifx\FamilyKeyState\FamilyKeyStateProcessed%

```

Setting the mode for invoking *Inkscape*. . .

```

69     \ifnum\svg@tempa<\thr@@\relax%
70       \let\svg@ink@mode\svg@tempa%
71     \else%

```

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

```

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

```

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

```

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

```

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

```

95   \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`.

```

96     \svg@quotes@remove[#{1}]{\svg@tempb}%
97     \def\svg@tempa##1-##2\@nil{%
98       \IfArgIsEmpty{##2}{\def\svg@tempb{}}{%
99         \def\svg@tempa####1-\@nil{\def\svg@tempb{-####1}}%
100        \svg@tempa##2\@nil%
101      }%
102      \edef\svg@tempa{\trim@spaces{##1}}%
103    }%
104    \edef\svg@tempb{%
105      \noexpand\svg@tempa\svg@tempb-\noexpand\@nil%
106    }%
107    \svg@tempb%
108    \if@svg@quotes@found%
109      \edef\svg@tempa{"\svg@tempa"}%
110    \fi%
111    \PackageWarning{svg}{%
112      Setting the executable%
113    \ifx\svg@tempb\@empty\else%

```

```

114     \space and associated options%
115     \fi%
116     \MessageBreak%
117     for Inkscape should be done with options\MessageBreak%
118     'inkscapeexe=\svg@tempa'%
119     \ifx\svg@tempb\@empty\else%
120         \MessageBreak and 'inkscapeopt=\svg@tempb'%
121     \fi.\MessageBreak%
122     Nevertheless, this was done by now anyway%
123 }%
124 \edef\svg@tempa{%
125     \noexpand\FamilyOptions{SVG}{inkscapeexe=\svg@tempa}%
126     \ifx\svg@tempb\@empty\else%
127         \noexpand\FamilyOptions{SVG}{inkscapeopt=\svg@tempb}%
128     \fi%
129 }%
130 \svg@tempa%
131 \fi%
132 \fi%
133 }

```

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

```

134 \DeclareOption{on}{\FamilyOptions{SVG}{inkscape=true}}
135 \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.

```

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

```

inkscapectex (opt.) This option controls whether the **Inkscape** export will be invoked with or without the  
latex (opt.) generation of a separate L<sup>A</sup>T<sub>E</sub>X file.

```

tex (opt.)
\svg@ink@latex
157 \newif\if@svg@ink@latex
158 \FamilyBoolKey{SVG}{inkscapectex}{@svg@ink@latex}
159 \FamilyBoolKey{SVG}{latex}{@svg@ink@latex}
160 \FamilyBoolKey{SVG}{tex}{@svg@ink@latex}

```

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

```

\svg@ink@area
161 \newcommand*\svg@ink@area{}

```

```

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

```

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

`inksapedensity` (opt.)

`\svg@ink@dpi`

```

175 \newcommand*{\svg@ink@dpi{}}
176 \let\svg@ink@dpi\relax
177 \DefineFamilyKey{SVG}{inksapedpi}{%
178   \FamilyKeyStateUnknownValue%
179   \svg@ifvalueisrelax{#1}{%
180     \let\svg@ink@dpi\relax%
181     \FamilyKeyStateProcessed%
182   }{%
183     \def\svg@tempa##1dpi##2\@nil{\def\svg@tempa{##1}}%
184     \lowercase{\svg@tempa#1dpi\@nil}%
185     \Ifnumber{\svg@tempa}{%
186       \edef\svg@ink@dpi{\svg@tempa}%
187       \FamilyKeyStateProcessed%
188     }{}%
189   }%
190 }
191 \DefineFamilyKey{SVG}{inksapedensity}{\FamilyOptions{SVG}{inksapedpi=#1}}

```

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

`\svg@ink@exe`

`inkscapeopt` (opt.)

`\svg@ink@opt`

```

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

```

### B.1.2. Setting input folder and file

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

```

202 \DefineFamilyKey{SVG}{svgpath}{%
203   \PackageWarning{svg}{%
204     The key ‘svgpath’ is deprecated. It’s recommended\MessageBreak%
205     to use ‘\string\svgpath’ instead%
206   }%
207 \ifx\svgpath\@undefined%
208   \AtEndOfPackage{\svgpath{#1}}}%
209 \else%
210   \svgpath{#1}}}%
211 \fi%

```



```

212 \FamilyKeyStateProcessed%
213 }

```

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

```

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

```

The extension should be in lower case letters.

```

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

```

Remove leading dots from the extension.

```

217 \svg@remove@leadingchar.\svg@file@ext%
218 }
219 \DefineFamilyKey{SVG}{extension}{\FamilyOptions{SVG}{svgextension=#1}}
220 \DefineFamilyKey{SVG}{ext}{\FamilyOptions{SVG}{svgextension=#1}}

```

### B.1.3. Setting output folder

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

```

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

```

### 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.

width (opt.) These options determine the size of the included graphics. The usage of `\relax` as value resets the respective option to the default behavior.

```

\svg@param@width
height (opt.)
\svg@param@width
distort (opt.)
keepaspectratio (opt.)
\if@svg@param@distort
scale (opt.)
\svg@param@scale
252 \newcommand*\svg@param@width{\z@}
253 \DefineFamilyKey{SVG}{width}{%
254   \FamilyKeyStateUnknownValue%
255   \svg@ifvalueisrelax{#1}{%
256     \renewcommand*\svg@param@width{\z@}%
257     \FamilyKeyStateProcessed%
258   }{%
259     \FamilySetLengthMacro{SVG}{width}{\svg@param@width}{#1}%
260     \ifx\FamilyKeyState\FamilyKeyStateProcessed%
261       \ifdim\svg@param@width<\z@\relax%
262         \FamilyKeyStateUnknownValue%
263       \fi%
264     \fi%
265   }%
266 }
267 \newcommand*\svg@param@height{\z@}
268 \DefineFamilyKey{SVG}{height}{%
269   \FamilyKeyStateUnknownValue%
270   \svg@ifvalueisrelax{#1}{%
271     \renewcommand*\svg@param@height{\z@}%
272     \FamilyKeyStateProcessed%
273   }{%
274     \FamilySetLengthMacro{SVG}{height}{\svg@param@height}{#1}%
275     \ifx\FamilyKeyState\FamilyKeyStateProcessed%
276       \ifdim\svg@param@height<\z@\relax%
277         \FamilyKeyStateUnknownValue%
278       \fi%
279     \fi%
280   }%
281 }
282 \newif\if@svg@param@distort
283 \FamilyBoolKey{SVG}{distort}{@svg@param@distort}
284 \DefineFamilyKey{SVG}{keepaspectratio}[true]{%
285   \FamilySetBool{SVG}{keepaspectratio}{@svg@tempswa}{#1}%
286   \ifx\FamilyKeyState\FamilyKeyStateProcessed%
287     \if@svg@tempswa%
288       \FamilyOptions{SVG}{distort=false}%
289     \else
290       \FamilyOptions{SVG}{distort=true}%
291     \fi%
292   \fi%
293 }
294 \newcommand*\svg@param@scale{1}
295 \DefineFamilyKey{SVG}{scale}{%
296   \FamilyKeyStateUnknownValue%
297   \svg@ifvalueisrelax{#1}{%
298     \renewcommand*\svg@param@scale{1}%
299     \FamilyKeyStateProcessed%
300   }{%
301     \Ifisdimension{#1\p@}{%
302       \ifdim\dimexpr#1\p@\relax>\z@\relax%
303         \renewcommand*\svg@param@scale{#1}%
304         \FamilyKeyStateProcessed%
305       \fi%
306     }{%
307   }%
308 }

pretex (opt.) For executing code right before or after the graphic inclusion, two hooks are defined.
\svg@param@pretex
apptex (opt.)
\svg@param@apptex
postex (opt.)
309 \newcommand*\svg@param@pretex{}
310 \let\svg@param@pretex\relax
311 \DefineFamilyKey{SVG}{pretex}{%

```

```

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

```

`lastpage` (opt.) For *Inkscape* 0.91 a bug concerning the L<sup>A</sup>T<sub>E</sub>X export has been reported (<https://bugs.launchpad.net/ubuntu/+source/inkscape/+bug/1417470>). Sometimes the L<sup>A</sup>T<sub>E</sub>X 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.

```

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

```

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

```

346 \newif\if@svg@draft
347 \FamilyBoolKey{SVG}{\draft}{@svg@draft}
348 \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<sup>A</sup>T<sub>E</sub>X, this is taken into account.

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

```

349 \newcommand*\svg@deactivate@dq{}
350 \AfterPackage+{babel}{%
351 \renewcommand*\svg@deactivate@dq{\bbl@deactivate{}}%
352 \providecommand*\bbl@deactivate[1]{}%
353 }

```

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

```

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

```

`\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.

`\svg@quotes@@remove`

```

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

```

`\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`.

`\svg@quotes@@check`

`\if@svg@quotes@found`

```

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

```

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

```

393 \newcommand*\svg@remove@leadingchar[2]{%
394   \begingroup%
395   \svg@sanitize@dq\svg@tempa{#2}%
396   \def\svg@tempb{%
397     \def\svg@tempa####1\@nil{\def\svg@tempa{####1}}%
398     \kernel@ifnextchar#1%
399     {\expandafter\svg@tempa\@gobble}%

```

```

400      {\svg@tempa}%
401    }%
402    \expandafter\svg@tempb\svg@tempa\@nil%
403    \edef\svg@tempb{%
404      \endgroup%
405      \def\noexpand#2{\svg@tempa}%
406    }%
407    \svg@tempb%
408  }

```

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

```

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

```

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

```

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

```

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

```

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

```

Finally, `\input@path` is set.

```

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

```

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

```

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

```

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

```

433     \ifx\svg@tempa\@empty\else%
434       \@svg@tempswatrue%
435       \svg@normalize@path{\svg@tempa}%
436       \expandafter\@tfor\expandafter\svg@tempb\expandafter:\expandafter=%
437       #1\do{%
438         \ifx\svg@tempa\svg@tempb%

```

```

439         \@svg@tempswafalse%
440         \@break@tfor%
441         \fi%
442     }%
443     \if@svg@tempswa%
444         \edef#1{#1{\svg@tempa}}%
445     \fi%
446 \fi%
447 }%
448 \fi%
449 }

```

`\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`.

```

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

```

`\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.

```

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

```

`\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

`\if@svg@file@found`  
`\svg@file@path`  
`\svg@file@name`  
`\svg@file@base`  
`\svg@file@suffix`



name are stored in `\svg@file@path` and `\svg@file@name` as well as the compound of both is saved in `\svg@file@base`.

```

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

A maybe given, unneeded file extension is removed.

```

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

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

```

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

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

```

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

`\svg@filename@parse` As the internal L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> command `\filename@parse` is not able to split a given file name containing quotes, `\svg@filename@parse` is defined to resolve this problem. The optional argument can be used to give a specific file extension, which should be searched within `\filename@ext`. If found at the very end, the previous part is appended to `\filename@base`.

```

527 \newcommand*{\svg@filename@parse[2][]}{%
528   \begingroup%
```

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

```

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

```

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

```

541 \ifx\filename@ext\relax\else%
542 \svg@quotes@remove{\filename@ext}%
543 \svg@extension@parse{#1}%
544 \ifsvg@quotes@found%
545 \edef\filename@base{\filename@base"}%
546 \fi%
547 \fi%

```

Quotes within `\filename@base` are normalized.

```

548 \svg@quotes@remove{\filename@base}%
549 \ifsvg@quotes@found%
550 \edef\filename@base{"\filename@base"%
551 \fi%

```

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

```

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

```

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

```

564 \newcommand*\svg@extension@parse[1]{%
565 \IfArgIsEmpty{#1}{}{%
566 \@expandtwoargs\Ifstr%
567 {\detokenize\expandafter{\filename@ext}}{\detokenize\expandafter{#1}}{}{%
568 \begingroup%

```

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

```

569 \edef\svg@tempa{%
570 \def\noexpand\svg@tempa{}%
571 \let\noexpand\svg@tempb\relax%

```

```

572         \noexpand\svg@extension@@parse%
573         \filename@ext.\noexpand\@nil#1\noexpand\@nil%
574     }%
575     \svg@tempa%
576     \edef\svg@tempa{%
577         \endgroup%

```

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

```

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

```

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

```

589 \newcommand*\svg@extension@@parse{}
590 \def\svg@extension@@parse#1.#2\@nil#3\@nil{%
591     \edef\svg@tempa{\svg@tempa.#1}%
592     \IfArgIsEmpty{#2}{-}{%
593         \Ifstr{\detokenize{#2}}{\detokenize{#3.}}{-}%

```

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

```

594         \edef\svg@tempb{#3}%
595     }{-}%
596     \svg@extension@@parse#2\@nil#3\@nil%
597 }%
598 }%
599 }

```

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

```

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

```

Collecting all considered path for the error message.

```

614         \edef\svg@tempb{#3}%
615         \Ifstr{\svg@tempb}{./}{\let\svg@tempb\@empty}{-}%
616         \ifx\svg@tempb\@empty%
617             \svg@set@input@path%
618         \else%
619             \svg@set@input@path[\svg@tempb]%
620         \fi%

```

```

621 \ifx\input@path\@undefined%
622 \def\svg@tempb{No additional path was given.}%
623 \else%
624 \def\svg@tempb{Following folders have additionally been searched:}%
625 \expandafter\@tfor\expandafter\svg@tempa\expandafter:\expandafter=%
626 \input@path\do{%
627 \edef\svg@tempb{\svg@tempb\noexpand\MessageBreak\svg@tempa}%
628 }%
629 \fi%

```

The error message itself.

```

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

```

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

```

646 \newcommand*\svg@iffilenewer[2]{\@gobbletwo}
647 \ifx\pdf@filemoddate\@undefined
648 \ifx\filemoddate\@undefined\else
649 \ifx\strcmp\@undefined\else
650 \renewcommand*\svg@iffilenewer[2]{%
651 \begingroup%
652 \edef\svg@tempa{\filemoddate{#1}}%
653 \edef\svg@tempb{\filemoddate{#2}}%
654 \ifnum\strcmp{\svg@tempa}{\svg@tempb}>\z@\relax%
655 \aftergroup\@firstoftwo%
656 \else%
657 \aftergroup\@secondoftwo%
658 \fi%
659 \endgroup%
660 }%
661 \fi
662 \fi
663 \else
664 \ifx\pdf@strcmp\@undefined\else
665 \renewcommand*\svg@iffilenewer[2]{%
666 \begingroup%
667 \edef\svg@tempa{\pdf@filemoddate{#1}}%
668 \edef\svg@tempb{\pdf@filemoddate{#2}}%
669 \ifnum\pdf@strcmp{\svg@tempa}{\svg@tempb}>\z@\relax%
670 \aftergroup\@firstoftwo%
671 \else%
672 \aftergroup\@secondoftwo%
673 \fi%
674 \endgroup%
675 }%
676 \fi
677 \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**.

```
678 \newcommand*\svg@local@param@set[1]{%  
679   \svg@local@param@use%  
680   \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`.

```
681   \Ifstr{\svg@ink@format}{png}{\FamilyOptions{SVG}{inkscapelatex=false}}{}}%
```

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

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

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

```
696 \*body>  
697 \DefineFamilyMember[.param]{SVG}  
698 </body>
```

### 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.

```
699 \newcommand*\svgsetup{\FamilyOptions{SVG}}  
700 \newcommand*\setsvg{\FamilyOptions{SVG}}
```

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

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

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

```
713 \newcommand*\includesvg[2] [] {%
714   \begingroup%
```

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

```
715   \svg@deprecated@param%
```

`inkscape` (param.) Most of the optional parameters have the same effect as the identically named options.  
`inkscapeformat` (param.) Only parameter `lastpage` is extended (see below). Moreover, there are some additional  
`inkscapelatex` (param.) parameters, which can only be used as optional argument for `\includesvg` (`angle` and  
`inkscapearea` (param.) `origin`) but not as an option. Now all parameters are set in local context (within a group).  
`inkscapedpi` (param.)  
`inkscapeopt` (param.) 716 `\svg@local@param@set{#1}%`  
`svgextension` (param.)

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

```
width (param.)
height (param.)
distort (param.) 717 \if@svg@ink@latex%
718 \edef\svg@file@suffi{_{\svg@file@ext-tex}%
scale (param.) 719 \else%
720 \edef\svg@file@suffi{_{\svg@file@ext-raw}%
pretex (param.) 721 \fi%
722 \onelevel@sanitize\svg@file@suffi%
apptex (param.)
draft (param.)
```

Searching all given paths for the relevant SVG file.

```
723 \svg@get@path{#2}{}%
724 \if@svg@file@found%
```

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

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

Include the resulting graphic file and maybe extract independent files.

```
736 \if@svg@file@found%
737 \svg@input{\svg@out@base}%
738 \svg@extract{\svg@out@base}%
739 \fi%
740 \else%
```

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

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

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

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

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

```

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

```

`\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.

```

764 \newcommand*{includeinkscape}[2] [] {%
765   \begingroup%

```

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

```

766   \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.

```

767   \svg@filename@parse{#2}%
768   \ifx\filename@ext\relax\else%
769     \svg@quotes@remove{\filename@ext}%
770     \expandafter\lowercase\expandafter{%
771       \expandafter\def\expandafter\filename@ext\expandafter{\filename@ext}%
772     }%
773     \def\svg@tempb##1_tex##2\@nil{%
774       \IfArgIsEmpty{##1}{\def\filename@ext{##1}}%
775       \Ifstr{##2}{_tex}{\@svg@tempswatruetrue}{\@svg@tempswafalse}%
776     }%
777     \@svg@tempswafalse%
778     \@tfor\svg@tempa:={pdf}{eps}{ps}{png}\do{%
779       \begingroup%
780         \expandafter\svg@tempb\filename@ext_tex\@nil%
781         \svg@extension@parse{\svg@tempa}%
782         \ifx\filename@ext\relax%
783           \def\svg@tempb{\endgroup}%
784         \else%
785           \edef\svg@tempb{%
786             \endgroup%
787             \noexpand\FamilyOptions{SVG}{inkscapeformat=\svg@tempa}%
788             \if@svg@tempswa%
789               \noexpand\FamilyOptions{SVG}{inkscapelatex=true}%
790             \fi%
791             \def\noexpand\filename@base{\filename@base}%
792             \def\noexpand\filename@ext{\filename@ext}%
793             \noexpand\@svg@tempswatruetrue%
794           }%
795         \fi%
796       \svg@tempb%

```

Break for loop, if valid extension was found.

```

797      \if@svg@tempswa%
798      \@break@tfor%
799      \fi%
800  }%

```

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

```

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

```

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

```

height (param.)
distort (param.) 805 \svg@local@param@set{#1}%
scale (param.)

```

Searching all given paths for the relevant PDF/EPS file.

```

pretex (param.)
apptex (param.) 806 \svg@get@path[\svg@ink@format]{\filename@area\filename@base}{\svg@out@path}%
draft (param.) 807 \if@svg@file@found%
lastpage (param.)

```

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

```

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

```

Include the resulting graphic file and maybe extract independent files.

```

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

```

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

```

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

```

## 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.

```

825 \newcommand*\svg@deprecated@param{%
826 \@svg@tempwafalse%
827 \ifx\svgwidth\@undefined\else%
828 \edef\svg@tempa{\noexpand\FamilyOptions{SVG}{width=\svgwidth}}%
829 \svg@tempa%
830 \@svg@tempwattrue%
831 \fi%
832 \ifx\svgscale\@undefined\else%

```



```

833 \edef\svg@tempa{\noexpand\FamilyOptions{SVG}{scale=\svgscale}}%
834 \svg@tempa%
835 \@svg@tempswatrue%
836 \fi%
837 \if@svg@tempswa%
838 \PackageWarning{svg}{%
839     You should specify the image size with parameters\MessageBreak%
840     'width' and 'height' or 'scale' instead of using\MessageBreak%
841     '\string\svgscale' or '\string\svgwidth'%
842 }%
843 \let\svgwidth\undefined%
844 \let\svgscale\undefined%
845 \fi%
846 }

```

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

```

847 \newif\if@svg@ink@run
848 \newcommand*\svg@ink@run{%
849     \ifnum\svg@ink@mode>\z@\relax%
850     \begingroup%

```

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

```

851     \@svg@ink@runtrue%
852     \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.

```

853     \svg@iffilenewer{\svg@file@base.\svg@file@ext}{\svg@out@base}{}%
854     \@svg@ink@runfalse%
855 }%

```

The same is true, when the associated L<sup>A</sup>T<sub>E</sub>X 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.

```

856     \if@svg@ink@latex%
857     \IfFileExists{\svg@out@base_tex}{%
858     \ifnum\pdf@shellescape=\@ne\relax\if@svg@ink@run%
859     \svg@iffilenewer{\svg@out@base_tex}{\svg@out@base}{%
860     \@svg@ink@runfalse%
861     \svg@quotes@remove[\svg@out@base]{\svg@tempa}%
862     \PackageWarning{svg}{%
863         Since the encountered filedate of file\MessageBreak%
864         '\svg@tempa_tex' is newer than \MessageBreak%
865         '\svg@tempa' it's supposed that\MessageBreak%
866         you customized this file. To avoid an accidental\MessageBreak%
867         overwriting of this file, the Inkscape export\MessageBreak%
868         won't be done. If you want to overwrite the\MessageBreak%
869         existing file please choose the parameter\MessageBreak%
870         'inkscape=force'%
871     }%
872     }{}%
873     \fi\fi%
874     }\@svg@ink@runtrue}%
875 \fi%
876 \fi%

```

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

```

877     \if@svg@ink@run%
878     \ifnum\pdf@shellescape=\@ne\relax%

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

`\svg@get@lastpage` This macro is used to circumvent the multiple pages bug for PDF files of **Inkscape** 0.91, when the the L<sup>A</sup>T<sub>E</sub>X export was enabled. For this purpose, the total page number is read from the PDF file.

```

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

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

```

964 \newcommand*\svg@wrn@scale{%
965   \ifdim\dimexpr\svg@param@scale\p@\relax=\p@\relax\else%
966     \@svg@tempswafalse%
967     \ifdim\svg@param@width>\z@\relax%
968       \@svg@tempswatrue%
969     \fi%
970     \ifdim\svg@param@height>\z@\relax%
971       \@svg@tempswatrue%
972     \fi%
973     \if@svg@tempswa%
974       \PackageWarning{svg}{%
975         The parameter 'scale' is only considered if neither\MessageBreak%
976         'width' nor 'height' are specified%
977       }%
978     \fi%
979   \fi%
980 }
```

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

```

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

```

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

```

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

```

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

```

1001 \svg@remove@leadingchar.\svg@tempa%
1002 \begingroup%
1003   \expandafter\filename@parse\expandafter{\svg@tempa}%
1004   \edef\svg@tempa{%
1005     \endgroup%
1006     \ifx\filename@ext\relax%
1007       \edef\noexpand\svg@tempa{\svg@tempa.\svg@tempb}%
1008     \else%
1009       \edef\noexpand\svg@tempa{{\svg@tempa}.\svg@tempb}%
1010     \fi%
1011   }%
1012 \svg@tempa%
1013 \edef\svg@tempb{#2}%

```

If the export with *Inkscape* was done with L<sup>A</sup>T<sub>E</sub>X 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`.

```

1014 \if@svg@ink@latex%
1015   \svg@patches{\svg@tempa}%
1016   \ifnum\value{svg@param@lastpage}=\z@%
1017     \expandafter\svg@get@lastpage\expandafter{\svg@tempb}%
1018   \fi%
1019   \edef\svg@tempa{%
1020     \ifx\svg@param@pretex\relax%
1021       \noexpand\svg@param@pretex%
1022     \fi%
1023     \noexpand\input{\svg@tempb_tex}%
1024     \ifx\svg@param@apptex\relax%
1025       \noexpand\svg@param@apptex%
1026     \fi%
1027   }%

```

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

```

1028   \if@svg@param@distort%

```

```

1029     \def\svg@tempb{\resizebox*{\svg@param@width}{\svg@param@height}}%
1030     \else%
1031     \let\svg@tempb\@firstofone%
1032     \fi%
1033     \sbox\svg@box{\svg@tempb{\svg@tempa}}%

```

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

```

1034     \ifdim\dimexpr\svg@param@angle\p\relax=\z@ \relax%
1035     \let\svg@tempb\@firstofone%
1036     \else%
1037     \edef\svg@tempb{%
1038         \noexpand\rotatebox[origin=\svg@param@origin]{\svg@param@angle}%
1039     }%
1040     \fi%
1041     \svg@tempb{\usebox\svg@box}%
1042     \else%

```

If the export with *Inkscape* was done without L<sup>A</sup>T<sub>E</sub>X support, the resulting graphic file will be included with `\includegraphics`.

```

1043     \svg@wrn@scale%
1044     \edef\svg@tempb{%
1045         draft\if@svg@draft\else=false\fi,%
1046         scale=\svg@param@scale,%
1047         keepaspectratio\if@svg@param@distort=false\fi%
1048     }%
1049     \ifdim\svg@param@height>\z@ \relax%
1050     \edef\svg@tempb{\svg@tempb,height=\svg@param@height}%
1051     \fi%
1052     \ifdim\svg@param@width>\z@ \relax%
1053     \edef\svg@tempb{\svg@tempb,width=\svg@param@width}%
1054     \fi%
1055     \ifdim\dimexpr\svg@param@angle\p\relax=\z@ \relax\else%
1056     \edef\svg@tempb{%
1057         \svg@tempb,origin=\svg@param@origin,angle=\svg@param@angle%
1058     }%
1059     \fi%
1060     \expandafter\includegraphics\expandafter[\svg@tempb]{\svg@tempa}%
1061     \fi%
1062 }

```

## B.6. Patches

`\svg@patches` For including the export results from *Inkscape* with L<sup>A</sup>T<sub>E</sub>X support enabled, there are some patches necessary for environment `picture` and `\includegraphics`. Those patches are done with `\svg@patches`.

```

1063 \newcommand*\svg@picture@savd{}
1064 \let\svg@picture@savd\picture
1065 \newcommand*\svg@includegraphics@savd{}
1066 \let\svg@includegraphics@savd\includegraphics
1067 \newcommand*\svg@patches[1]{%
1068     \let\picture\svg@picture@patched%
1069     \let\includegraphics\svg@includegraphics@patched%
1070     \edef\svg@includegraphics@file{#1}%
1071 }

```

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

```

1072 \newcommand*\svg@picture@patched{}
1073 \newcommand*\svg@picture@patched{}
1074 \long\def\svg@picture@patched#1{\svg@picture@patched@#1}

```

```

1075 \def\svg@pictur@patched@(#1,#2){%
1076   \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.

```

1077   \ifdim\svg@param@height>\z@\relax%
1078     \Gscale@div\svg@tempa{#1\p@}{#2\p@}%
1079     \setlength\unitlength{\svg@param@height}%
1080     \setlength\unitlength{\svg@tempa\unitlength}%
1081   \ifdim\svg@param@width>\z@\relax%
1082     \ifdim\unitlength>\svg@param@width\relax%
1083       \setlength\unitlength{\svg@param@width}%
1084     \fi%
1085   \fi%
1086 \else%

```

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

```

1087   \ifdim\svg@param@width>\z@\relax%
1088     \setlength\unitlength{\svg@param@width}%
1089   \else%
1090     \setlength\unitlength{\svg@param@scale\unitlength}%
1091   \fi%
1092 \fi%

```

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

```

1093   \svg@picture@saved(#1,#2)%
1094 }

```

`\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.

```

1095 \DefineFamily{SVGpatch}
1096 \DefineFamilyMember{SVGpatch}
1097 \newcounter{svg@param@currpage}
1098 \setcounter{svg@param@currpage}{\m@ne}
1099 \FamilyCounterKey{SVGpatch}{page}{svg@param@currpage}
1100 \DefineFamilyKey{SVGpatch}{width}{\FamilyKeyStateProcessed}
1101 \newcommand*\svg@includegraphics@file{}
1102 \newcommand*\svg@includegraphics@patched[2][]{%
1103   \FamilyOptions{SVGpatch}{#1}%

```

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

```

1104   \ifnum\value{svg@param@lastpage}<\z@\relax%
1105     \FamilySetCounter{SVGpatch}{page}{svg@param@currpage}{%
1106       \the\value{svg@param@lastpage}%
1107     }%
1108   \fi%

```

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

```

1109   \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.

```
1110 \svg@includegraphics@saved[{#1}]{\svg@includegraphics@file}%
1111 \fi%
1112 }
```

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

### C.1. Options

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

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

```
1113 (*base)
1114 \DefineFamilyMember[.dummy]{SVG}
1115 \newcommand*{\svg@dummy@key[2][]}{%
1116   \@ifpackageloaded{svg-extract}{}{%
1117     \IfArgIsEmpty{#1}{%
1118       \DefineFamilyKey[.dummy]{SVG}{#2}{%
1119         \PackageWarning{svg}{%
1120           The option key '#2' can only\MessageBreak%
1121           be used with package 'svg-extract', but\MessageBreak%
1122           you didn't load it%
1123         }%
1124         \FamilyKeyStateProcessed%
1125       }%
1126     }{%
1127       \DefineFamilyKey[.dummy]{SVG}{#2}[{#1}]{%
1128         \PackageWarning{svg}{%
1129           The option key '#2' can only\MessageBreak%
1130           be used with package 'svg-extract', but\MessageBreak%
1131           you didn't load it%
1132         }%
1133         \FamilyKeyStateProcessed%
1134       }%
1135     }%
1136   }
```

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

```
1136 \AfterPackage{svg-extract}{\RelaxFamilyKey[.dummy]{SVG}{#2}}%
1137 }%
1138 }
1139 </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.

`\ifsvgx@run`

```
1140 (*base)
1141 \svg@dummy@key[true]{extract}
1142 </base>
1143 (*extract)
1144 \newif\ifsvgx@run
1145 \DefineFamilyKey{SVG}{extract}[true]{%
1146   \lowercase{\def\svg@tempa{#1}}%
1147   \FamilySetNumerical{SVG}{extract}{\svg@tempa}{%
1148     \ifsvgx@run
1149       \svg@tempa
1150     \else
1151       \relax
1152     \fi
1153   }%
1154 }
```

```

1148 {false}{0},{off}{0},{no}{0},%
1149 {true}{1},{on}{1},{yes}{1},{onlynewer}{1},{newer}{1},%
1150 {overwrite}{1},{force}{1},{forced}{1},%
1151 {pdf}{2},{eps}{3},{ps}{4}%
1152 }\svg@tempa}%
1153 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1154 \ifcase\svg@tempa\relax% false
1155 \svgx@runfalse%
1156 \or% true
1157 \svgx@runtrue%
1158 \or% pdf
1159 \FamilyOptions{SVG}{extractformat=pdf}%
1160 \or% eps
1161 \FamilyOptions{SVG}{extractformat=eps}%
1162 \or% ps
1163 \FamilyOptions{SVG}{extractformat=ps}%
1164 \fi%
1165 \fi%
1166 }
1167 \end{extract}

```

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

```

1168 \begin{extract}
1169 \DeclareOption{on}{\FamilyOptions{SVG}{extract=true}}
1170 \DeclareOption{off}{\FamilyOptions{SVG}{extract=false}}
1171 \end{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.)
1172 \begin{base}
1173 \svg@dumkey{extractformat}
1174 \svg@dumkey[true]{pdf}
1175 \svg@dumkey[true]{eps}
1176 \end{base}
1177 \begin{extract}
1178 \newcommand*\svgx@format{pdf}
1179 \ifxetex\else\ifpdf\else
1180 \renewcommand*\svgx@format{eps}
1181 \fi\fi
1182 \DefineFamilyKey{SVG}{extractformat}{%
1183 \lowercase{\edef\svgx@format{#1}}%
1184 \FamilyKeyStateProcessed%
1185 }
1186 \DefineFamilyKey{SVG}{pdf}[true]{%
1187 \FamilySetBool{SVG}{pdf}{\svg@tempa}{#1}%
1188 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1189 \if\svg@tempa%
1190 \svgx@ifinlist{pdf}{\svgx@format}{%
1191 \edef\svgx@format{\svgx@format,pdf}%
1192 }%
1193 \svg@deprecated@key{pdf}{extractformat={\svgx@format}}%
1194 \else%
1195 \FamilyKeyStateUnknownValue%
1196 \fi%
1197 \fi%
1198 }
1199 \DefineFamilyKey{SVG}{eps}[true]{%
1200 \FamilySetBool{SVG}{eps}{\svg@tempa}{#1}%
1201 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1202 \if\svg@tempa%
1203 \svgx@ifinlist{eps}{\svgx@format}{%
1204 \edef\svgx@format{\svgx@format,eps}%

```



	<pre> 1205     }% 1206     \svg@deprecated@key{eps}{extractformat={\svgx@format}}% 1207     \else% 1208         \FamilyKeyStateUnknownValue% 1209     \fi% 1210 \fi% 1211 } 1212 \end{extract} </pre>	
<code>extractpreamble (opt.)</code>	For the extraction process, a preamble is necessary for a separate auxiliary L <sup>A</sup> T <sub>E</sub> X file.	
<code>preamble (opt.)</code>	By default, the preamble of the main document is used, which end is detected at	
<code>\svgx@preamble</code>	<code>\begin{document}</code> .	
<code>extractpreambleend (opt.)</code>		
<code>end (opt.)</code>	<pre> 1213 \begin{base} 1214 \svg@dummy@key{extractpreamble} 1215 \svg@dummy@key{preamble} 1216 \svg@dummy@key{extractpreambleend} 1217 \svg@dummy@key{end} 1218 \end{base} 1219 \begin{extract} 1220 \newcommand*\svgx@preamble{\jobname.\svgx@latex@ext}% 1221 \DefineFamilyKey{SVG}{extractpreamble}{% 1222     \renewcommand*\svgx@preamble{#1}% 1223     \FamilyKeyStateProcessed% 1224 } 1225 \DefineFamilyKey{SVG}{preamble}{% 1226     \svg@deprecated@key[svg-extract]{preamble=#1}{extractpreamble=#1}% 1227 } 1228 \newcommand*\svgx@endpreamble{} 1229 \expandafter\def\expandafter\svgx@endpreamble\expandafter{% 1230     \csname begin\endcsname{document}% 1231 } 1232 \DefineFamilyKey{SVG}{extractpreambleend}{% 1233     \renewcommand*\svgx@endpreamble{#1}% 1234     \FamilyKeyStateProcessed% 1235 } 1236 \DefineFamilyKey{SVG}{end}{% 1237     \svg@deprecated@key[svg-extract]{end=#1}{extractpreambleend=#1}% 1238 } 1239 \end{extract} </pre>	
<code>\svgx@endpreamble</code>		
<code>extractruns (opt.)</code>	With this option, the number of L <sup>A</sup> T <sub>E</sub> X runs for the separate auxiliary file can be set.	
<code>svgx@runs (counter)</code>	<pre> 1240 \begin{base} 1241 \svg@dummy@key{extractruns} 1242 \end{base} 1243 \begin{extract} 1244 \newcounter{svgx@runs} 1245 \DefineFamilyKey{SVG}{extractruns}{% 1246     \FamilySetCounter{SVG}{extractruns}{svgx@runs}{#1}% 1247     \ifx\FamilyKeyState\FamilyKeyStateProcessed% 1248         \ifnum\value{svgx@runs}&lt;\@ne\relax% 1249             \PackageWarning{svg-extract}{% 1250                 The count for runs has to be at least one% 1251             }% 1252             \FamilySetCounter{SVG}{extractruns}{svgx@runs}{\@ne}% 1253         \fi% 1254     \fi% 1255 } 1256 \end{extract} </pre>	
<code>latexexe (opt.)</code>	The command and facultative options for the L <sup>A</sup> T <sub>E</sub> X call of the separate auxiliary file. The	
<code>pdflatex (opt.)</code>	default is set according to the currently used compiler.	
<code>\svgx@latex@exe</code>		
<code>latexext (opt.)</code>	<pre> 1257 \begin{base} </pre>	
<code>\svgx@latex@ext</code>		
<code>latexopt (opt.)</code>	File II: svg.dtx Date: 2020/01/13 v2.02e	
<code>\svgx@latex@opt</code>		

```

1258 \svg@dummy@key{latexexe}
1259 \svg@dummy@key{pdflatex}
1260 \svg@dummy@key{latexext}
1261 \svg@dummy@key{latexopt}
1262 </base>
1263 <*extract>
1264 \ifxetex
1265   \newcommand*\svgx@latex@exe{xelatex}
1266 \else\ifluatex
1267   \ifpdf
1268     \newcommand*\svgx@latex@exe{lualatex}
1269   \else
1270     \newcommand*\svgx@latex@exe{lualatex --output-format=dvi}
1271   \fi
1272 \else\ifpdf
1273   \newcommand*\svgx@latex@exe{pdflatex}
1274 \else
1275   \newcommand*\svgx@latex@exe{latex}
1276 \fi\fi\fi
1277 \DefineFamilyKey{SVG}{latexexe}{%
1278   \renewcommand*\svgx@latex@exe{#1}%
1279   \FamilyKeyStateProcessed%
1280 }
1281 \DefineFamilyKey{SVG}{pdflatex}{%
1282   \svg@deprecated@key[svg-extract]{pdflatex=#1}{latexexe=#1}%
1283 }
1284 \newcommand*\svgx@latex@ext{tex}
1285 \DefineFamilyKey{SVG}{latexext}{%
1286   \renewcommand*\svgx@latex@ext{#1}%
1287   \FamilyKeyStateProcessed%
1288 }
1289 \newcommand*\svgx@latex@opt{}
1290 \DefineFamilyKey{SVG}{latexopt}{%
1291   \renewcommand*\svgx@latex@opt{#1}%
1292   \FamilyKeyStateProcessed%
1293 }
1294 </extract>

```

dvipsopt (opt.)  
 \svgx@dvips@exe  
 \svgx@dvips@opt  
 pstoeptsopt (opt.)  
 \svgx@pstoeptsopt@exe  
 \svgx@pstoeptsopt@opt  
 pstopdfopt (opt.)  
 \svgx@pstopdf@exe  
 \svgx@pstopdf@opt  
 pdftoeptsopt (opt.)  
 \svgx@pdftoeptsopt@exe  
 \svgx@pdftoeptsopt@opt  
 pdftops (opt.)  
 \svgx@pdftops@opt

Options and macros for calling convert commands, which are supplied by most L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> distributions. These are used to generate all files, which are supported by option `extractformat`, as they don't need an additional application.

```

1295 <*base>
1296 \svg@dummy@key{dvipsopt}
1297 \svg@dummy@key{pstoeptsopt}
1298 \svg@dummy@key{pstopdfopt}
1299 \svg@dummy@key{pdftoeptsopt}
1300 \svg@dummy@key{pdftops}
1301 \svg@dummy@key{pdftops}
1302 </base>
1303 <*extract>
1304 \newcommand*\svgx@dvips@exe{dvips}
1305 \newcommand*\svgx@dvips@opt{}
1306 \DefineFamilyKey{SVG}{dvipsopt}{%
1307   \renewcommand*\svgx@dvips@opt{#1}%
1308   \FamilyKeyStateProcessed%
1309 }
1310 \newcommand*\svgx@pstoeptsopt@exe{ps2eps}
1311 \newcommand*\svgx@pstoeptsopt@opt{-B -C}
1312 \DefineFamilyKey{SVG}{pstoeptsopt}{%
1313   \renewcommand*\svgx@pstoeptsopt@opt{#1}%
1314   \FamilyKeyStateProcessed%
1315 }
1316 \newcommand*\svgx@pstopdf@exe{ps2pdf}
1317 \newcommand*\svgx@pstopdf@opt{}

```

```

1318 \DefineFamilyKey{SVG}{pstopdfopt}{%
1319   \renewcommand*{svgx@pstopdf@opt{#1}%
1320   \FamilyKeyStateProcessed%
1321 }
1322 \newcommand*{svgx@pdfstopdf@exe{pdfstopdf -eps}
1323 \newcommand*{svgx@pdfstopdf@opt{}}
1324 \DefineFamilyKey{SVG}{pdfstopdfsopt}{%
1325   \renewcommand*{svgx@pdfstopdfsopt@opt{#1}%
1326   \FamilyKeyStateProcessed%
1327 }
1328 \newcommand*{svgx@pdfstopdf@exe{pdfstopdf}
1329 \newcommand*{svgx@pdfstopdf@opt{}}
1330 \DefineFamilyKey{SVG}{pdfstopdfsopt}{%
1331   \renewcommand*{svgx@pdfstopdfsopt@opt{#1}%
1332   \FamilyKeyStateProcessed%
1333 }
1334 \DefineFamilyKey{SVG}{pdfstopdf}{%
1335   \PackageWarning{#1}{%
1336     The option key 'pdfstopdf' is deprecated.\MessageBreak%
1337     You should use either 'pdfstopdfsopt' or\MessageBreak%
1338     'pdfstopdfsopt' instead. See the manual for\MessageBreak%
1339     more. Nothing was done%
1340   }%
1341   \FamilyKeyStateProcessed%
1342 }
1343 \end{extract}

```

### C.1.2. Invoking external application for graphic conversion

Besides the use of a conversion tool supplied by L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>, 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 used. *ImageMagick* is set by default.

```

\if@svgx@cnv@run
\svgx@cnv@cmd
1344 (*base)
1345 \svg@dummy@key[true]{convert}
1346 \end{base}
1347 (*extract)
1348 \newif\if@svgx@cnv@run
1349 \newcommand*{svgx@cnv@cmd}{
1350 \DefineFamilyKey{SVG}{convert}[true]{%
1351   \FamilySetNumerical{SVG}{convert}{svg@tempa}{%
1352     {false}{0},{off}{0},{no}{0},%
1353     {true}{1},{on}{1},{yes}{1},{onlynewer}{1},{newer}{1},%
1354     {overwrite}{1},{force}{1},{forced}{1},%
1355     {magick}{2},{imagemagick}{2},{convert}{2},%
1356     {gs}{3},{ghostscript}{3},%
1357     {gs64}{4},{ghostscript64}{4},%
1358     {gs32}{5},{ghostscript32}{5}%
1359   }{#1}%
1360   \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1361     \ifcase\svg@tempa\relax% false
1362       \svgx@cnv@runfalse%
1363     \or% true
1364       \svgx@cnv@runtrue%
1365     \or% magick
1366       \svgx@cnv@runtrue%
1367     \renewcommand*{svgx@cnv@cmd}{\svgx@magick@cmd}%
1368     \or% gs
1369       \svgx@cnv@runtrue%
1370     \renewcommand*{svgx@cnv@cmd}{\svgx@gs@cmd}%
1371     \or% gs64
1372       \svgx@cnv@runtrue%

```

```

1373     \renewcommand*\svgx@cnv@cmd{\svgx@gs@cmd}%
1374     \svgx@onlywindows{%
1375       \renewcommand*\svgx@gs@exe{gswin64c}%
1376     }%
1377   \or% gs32
1378     \@svgx@cnv@runtrue%
1379     \renewcommand*\svgx@cnv@cmd{\svgx@gs@cmd}%
1380     \svgx@onlywindows{%
1381       \renewcommand*\svgx@gs@exe{gswin32c}%
1382     }%
1383   \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.

```

1384   \else%

```

Same doing like with option `inkscape`.

```

1385   \def\svg@tempa##1-##2\@nil{%
1386     \IfArgIsEmpty{##2}{\def\svg@tempb{}}{%
1387       \def\svg@tempa##1####1\@nil{\def\svg@tempb{####1}}%
1388       \svg@tempa#1\@nil%
1389     }%
1390     \def\svg@tempa{##1}%
1391   }%
1392   \svg@tempa#1-\@nil%
1393   \PackageWarning{svg-extract}{%
1394     Setting the executable%
1395     \ifx\svg@tempb\@empty\else%
1396       \space and associated options%
1397     \fi%
1398     \MessageBreak%
1399     for ImageMagick should be done with options\MessageBreak%
1400     'magickexe=\svg@tempa'%
1401     \ifx\svg@tempb\@empty\else%
1402       \MessageBreak and 'magicksetting' and/or 'magickoperator'%
1403     \fi.\MessageBreak%
1404     Nevertheless, this was done by now%
1405     \ifx\svg@tempb\@empty\else%
1406       , whereby \MessageBreak 'magicksetting=\svg@tempb' was used%
1407     \fi%
1408   }%
1409   \FamilyOptions{SVG}{convert=magick}%
1410   \edef\svg@tempa{%
1411     \noexpand\FamilyOptions{SVG}{magickexe=\svg@tempa}%
1412     \ifx\svg@tempb\@empty\else%
1413       \noexpand\FamilyOptions{SVG}{magicksetting=\svg@tempb}%
1414     \fi%
1415   }%
1416   \svg@tempa%
1417   \fi%
1418 }
1419 </extract>

```

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

`\svgx@cnv@format`  
`png` (opt.)

```

1420 <{*base>
1421 \svg@dumy@key{convertformat}
1422 \svg@dumy@key[true]{png}
1423 </base>
1424 <{*extract>
1425 \newcommand*\svgx@cnv@format{png}
1426 \DefineFamilyKey{SVG}{convertformat}{%

```

```

1427 \lowercase{\edef\svgx@cnv@format{#1}}%
1428 \ifx\svgx@cnv@format\empty\else%
1429 \@svgx@cnv@runtrue%
1430 \fi%
1431 \FamilyKeyStateProcessed%
1432 }
1433 \DefineFamilyKey{SVG}{png}[true]{%
1434 \FamilySetBool{SVG}{png}{\svg@tempswa}{#1}%
1435 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1436 \if\svg@tempswa%
1437 \svgx@ifinlist{png}{\svgx@cnv@format}{}%
1438 \edef\svgx@cnv@format{\svgx@cnv@format,png}%
1439 }%
1440 \svgx@deprecated@key{png}{convertformat={\svgx@cnv@format}}%
1441 \else%
1442 \FamilyKeyStateUnknownValue%
1443 \fi%
1444 \fi%
1445 }
1446 </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`.

```

1447 (*base)
1448 \svg@dummy@key{convertdpi}
1449 \svg@dummy@key{convertdensity}
1450 </base>
1451 (*extract)
1452 \newcommand*\svgx@cnv@dpi{}
1453 \let\svgx@cnv@dpi\relax
1454 \DefineFamilyKey{SVG}{convertdpi}{%
1455 \FamilyKeyStateUnknownValue%
1456 \svgx@ifkeyandval{#1}{%
1457 \svgx@cnv@get@dpi{##2}%
1458 \ifx\svg@tempa\relax\else%
1459 \expandafter\edef\csname svgx@cnv@dpi@##1\endcsname{\svg@tempa}%
1460 \FamilyKeyStateProcessed%
1461 \fi%
1462 }{%
1463 \svgx@cnv@get@dpi{##1}%
1464 \ifx\svg@tempa\relax\else%
1465 \edef\svgx@cnv@dpi{\svg@tempa}%
1466 \FamilyKeyStateProcessed%
1467 \fi%
1468 }%
1469 }
1470 \DefineFamilyKey{SVG}{convertdensity}{\FamilyOptions{SVG}{convertdpi=#1}}
1471 </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.

```

1472 (*base)
1473 \svg@dummy@key{magickexe}
1474 \svg@dummy@key{magicksetting}
1475 \svg@dummy@key{magickoperator}
1476 </base>
1477 (*extract)
1478 \newcommand*\svgx@magick@exe{}

```

```

1479 \DefineFamilyKey{SVG}{magickexe}{%
1480   \renewcommand*{svgx@magick@exe{#1}}%
1481   \FamilyKeyStateProcessed%
1482 }
1483 \newcommand*{svgx@magick@set{}}
1484 \DefineFamilyKey{SVG}{magicksetting}{%
1485   \svgx@setformatkey{#1}{svgx@magick@set}%
1486   \FamilyKeyStateProcessed%
1487 }
1488 \newcommand*{svgx@magick@opr{}}
1489 \DefineFamilyKey{SVG}{magickoperator}{%
1490   \svgx@setformatkey{#1}{svgx@magick@opr}%
1491   \FamilyKeyStateProcessed%
1492 }
1493 </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.

`svgx@gs@exe`

`gsopt` (opt.) It can either be set for all or a specific output format just like `gsopt` in the same manner like option `convertdpi`.

`svgx@gs@opt`

`gsdevice` (opt.)

`svgx@gs@device`

```

1494 <*base>
1495 \svg@dummy@key{gsexe}
1496 \svg@dummy@key{gsopt}
1497 \svg@dummy@key{gsdevice}
1498 </base>
1499 <*extract>
1500 \newcommand*{svgx@gs@exe{}}
1501 \DefineFamilyKey{SVG}{gsexe}{%
1502   \renewcommand*{svgx@gs@exe{#1}}%
1503   \FamilyKeyStateProcessed%
1504 }
1505 \newcommand*{svgx@gs@opt{}}
1506 \DefineFamilyKey{SVG}{gsopt}{%
1507   \svgx@setformatkey{#1}{svgx@gs@opt}%
1508   \FamilyKeyStateProcessed%
1509 }
1510 \newcommand*{svgx@gs@device{}}
1511 \DefineFamilyKey{SVG}{gsdevice}{%
1512   \svgx@setformatkey{#1}{svgx@gs@device}%
1513   \FamilyKeyStateProcessed%
1514 }
1515 </extract>

```

### C.1.3. Setting output folder

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

`path` (opt.)

`extractname` (opt.)

`name` (opt.)

`svgx@out@path`

`svgx@out@name`

`if@svgx@out@sec`

`svgx@out@count` (counter)

```

1516 <*base>
1517 \svg@dummy@key{extractpath}
1518 \svg@dummy@key{path}
1519 \svg@dummy@key{extractname}
1520 \svg@dummy@key{name}
1521 </base>
1522 <*extract>
1523 \newcommand*{svgx@out@path{}}
1524 \DefineFamilyKey{SVG}{extractpath}{%
1525   \svg@sanitize@dq\svg@tempb{#1}%
1526   \FamilySetNumerical{SVG}{extractpath}{svg@tempa}{%
1527     {svgpath}{0},{svgdir}{0},%
1528     {svgsubpath}{1},{svgsubdir}{1},%
1529     {basepath}{2},{basedir}{2},{jobpath}{2},{jobdir}{2},%

```

```

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

```

### C.1.4. Options for the extraction of graphics

<code>extractwidth</code> (opt.)	For graphic extraction, the given settings regarding the size for inclusion can be overwritten with these options. Using <code>\relax</code> as value leads to resetting an option as unset, regardless of what was previously given. The value <code>inherit</code> means, that the actual option for including is used for extraction as well. This is the default setting.
<code>\svgx@param@width</code>	
<code>extractheight</code> (opt.)	
<code>\svgx@param@width</code>	
<code>extractdistort</code> (opt.)	
<code>extractkeepaspectratio</code> (opt.)	1585 <code>\base</code>
<code>\svgx@param@distort</code>	
<code>extractscale</code> (opt.)	
<code>\svgx@param@scale</code>	

```

1586 \svg@dummy@key{extractwidth}
1587 \svg@dummy@key{extractheight}
1588 \svg@dummy@key{extractdistort}
1589 \svg@dummy@key{extractkeepaspectratio}
1590 \svg@dummy@key{extractscale}
1591 </base>
1592 <*extract>
1593 \newcommand*\svgx@param@width{\svg@param@width}
1594 \DefineFamilyKey{SVG}{extractwidth}{%
1595   \FamilyKeyStateUnknownValue%
1596   \svg@ifvalueisrelax{#1}{%
1597     \renewcommand*\svgx@param@width{\z@}%
1598     \FamilyKeyStateProcessed%
1599   }{%
1600     \Ifstr{#1}{inherit}{%
1601       \renewcommand*\svgx@param@width{\svg@param@width}%
1602       \FamilyKeyStateProcessed%
1603     }{%
1604       \FamilySetLengthMacro{SVG}{extractwidth}{\svgx@param@width}{#1}%
1605       \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1606         \ifdim\svgx@param@width<\z@\relax%
1607           \FamilyKeyStateUnknownValue%
1608         \fi%
1609       \fi%
1610     }%
1611   }%
1612 }
1613 \newcommand*\svgx@param@height{\svg@param@height}
1614 \DefineFamilyKey{SVG}{extractheight}{%
1615   \FamilyKeyStateUnknownValue%
1616   \svg@ifvalueisrelax{#1}{%
1617     \renewcommand*\svgx@param@height{\z@}%
1618     \FamilyKeyStateProcessed%
1619   }{%
1620     \Ifstr{#1}{inherit}{%
1621       \renewcommand*\svgx@param@height{\svg@param@height}%
1622       \FamilyKeyStateProcessed%
1623     }{%
1624       \FamilySetLengthMacro{SVG}{extractheight}{\svgx@param@height}{#1}%
1625       \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1626         \ifdim\svgx@param@height<\z@\relax%
1627           \FamilyKeyStateUnknownValue%
1628         \fi%
1629       \fi%
1630     }%
1631   }%
1632 }
1633 \newif\if@svgx@param@distort
1634 \DefineFamilyKey{SVG}{extractdistort}[true]{%
1635   \FamilyKeyStateUnknownValue%
1636   \svg@ifvalueisrelax{#1}{%
1637     \@svgx@param@distortfalse%
1638     \FamilyKeyStateProcessed%
1639   }{%
1640     \Ifstr{#1}{inherit}{%
1641       \renewcommand*\if@svgx@param@distort{\if@svg@param@distort}%
1642       \FamilyKeyStateProcessed%
1643     }{%
1644       \FamilySetBool{SVG}{extractdistort}{@svgx@param@distort}{#1}%
1645     }%
1646   }%
1647 }
1648 \DefineFamilyKey{SVG}{extractkeepaspectratio}[true]{%
1649   \FamilySetBool{SVG}{extractkeepaspectratio}{@svg@tempswa}{#1}%
1650   \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1651     \if@svg@tempswa%

```



```

1652 \FamilyOptions{SVG}{extractdistort=false}%
1653 \else
1654 \FamilyOptions{SVG}{extractdistort=true}%
1655 \fi%
1656 \else%
1657 \FamilyOptions{SVG}{extractdistort=#1}%
1658 \fi%
1659 }
1660 \newcommand*{svgx@param@scale}{\svg@param@scale}
1661 \DefineFamilyKey{SVG}{extractscale}{%
1662 \FamilyKeyStateUnknownValue%
1663 \svg@ifvalueisrelax{#1}{%
1664 \renewcommand*{svgx@param@scale}{1}%
1665 \FamilyKeyStateProcessed%
1666 }{%
1667 \Ifstr{#1}{inherit}{%
1668 \renewcommand*{svgx@param@scale}{\svg@param@scale}%
1669 \FamilyKeyStateProcessed%
1670 }{%
1671 \Ifisdimension{#1\p@}{%
1672 \ifdim\dimexpr#1\p@>\z@>\relax%
1673 \renewcommand*{svgx@param@scale}{#1}%
1674 \FamilyKeyStateProcessed%
1675 \fi%
1676 }{%
1677 }%
1678 }%
1679 }
1680 </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.)
1681 < *base>
1682 \svg@dummys@key{extractpretex}
1683 \svg@dummys@key{extractapptex}
1684 \svg@dummys@key{extractpostex}
1685 < /base>
1686 < *extract>
1687 \newcommand*{svgx@param@pretex}{\svg@param@pretex}
1688 \DefineFamilyKey{SVG}{extractpretex}{%
1689 \svg@ifvalueisrelax{#1}{%
1690 \let\svgx@param@pretex\relax%
1691 }{%
1692 \Ifstr{#1}{inherit}{%
1693 \renewcommand*{svgx@param@pretex}{\svg@param@pretex}%
1694 }{%
1695 \renewcommand*{svgx@param@pretex}{#1}%
1696 }%
1697 }%
1698 \FamilyKeyStateProcessed%
1699 }
1700 \newcommand*{svgx@param@apptex}{\svg@param@apptex}
1701 \DefineFamilyKey{SVG}{extractapptex}{%
1702 \svg@ifvalueisrelax{#1}{%
1703 \let\svgx@param@apptex\relax%
1704 }{%
1705 \Ifstr{#1}{inherit}{%
1706 \renewcommand*{svgx@param@apptex}{\svg@param@apptex}%
1707 }{%
1708 \renewcommand*{svgx@param@apptex}{#1}%
1709 }%
1710 }%
1711 \FamilyKeyStateProcessed%
1712 }
1713 \DefineFamilyKey{SVG}{extractpostex}{%

```

```

1714 \svg@deprecated@key[svg-extract]{extractpostex=#1}{extractapptex=#1}%
1715 }
1716 </extract>

```

### C.1.5. Miscellaneous options

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

**\svgx@clean**

```

1717 < *base>
1718 \svg@dummy@key[true]{clean}
1719 \svg@dummy@key[true]{clear}
1720 </base>
1721 < *extract>
1722 \newcommand*\svgx@clean{}
1723 \DefineFamilyKey{SVG}{clean}[true]{%
1724 \FamilySetBool{SVG}{clean}{@svg@tempswa}{#1}%
1725 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1726 \if@svg@tempswa%
1727 \renewcommand*\svgx@clean{log,aux,dvi,out,ps,eps,pdf,\svgx@latex@ext}%
1728 \else%
1729 \renewcommand*\svgx@clean{}%
1730 \fi%
1731 \else%
1732 \renewcommand*\svgx@clean{#1}%
1733 \FamilyKeyStateProcessed%
1734 \fi%
1735 }
1736 \DefineFamilyKey{SVG}{clear}{\FamilyOptions{SVG}{clean=#1}}
1737 </extract>

```

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

```

1738 < *base>
1739 \svg@dummy@key[true]{exclude}
1740 </base>
1741 < *extract>
1742 \DefineFamilyKey{SVG}{exclude}[true]{%
1743 \FamilySetBool{SVG}{exclude}{@svg@tempswa}{#1}%
1744 \ifx\FamilyKeyState\FamilyKeyStateProcessed%
1745 \if@svg@tempswa%
1746 \renewcommand*\svg@input[2][]{%
1747 \if@svgx@run\else%
1748 \PackageWarning{svg-extract}{%
1749 The image ‘##2’ was\MessageBreak%
1750 neither extracted nor included%
1751 }%
1752 \fi%
1753 }%
1754 \else%
1755 \renewcommand*\svg@input{\svg@input}%
1756 \fi%
1757 \fi%
1758 }
1759 </extract>

```

## C.2. User commands

<b>\includesvg</b>	The parameters <b>angle</b> and <b>origin</b> are defined as pendants to the keys provided by <b>\includegraphics</b> .
<b>extract</b> (param.)	
<b>extractpreamble</b> (param.)	
<b>extractformat</b> (param.)	
<b>extractwidth</b> (param.)	
<b>extractheight</b> (param.)	
<b>extractdistort</b> (param.)	
<b>extractscale</b> (param.)	
<b>extractangle</b> (param.)	
<b>extractpretex</b> (param.)	

```

1760 <*extract>
1761 \newcommand*\svgx@param@angle{0}
1762 \svg@local@param@def{%
1763   \DefineFamilyKey[.param]{SVG}{extractangle}{%
1764     \FamilyKeyStateUnknownValue%
1765     \svg@ifvalueisrelax{#1}{%
1766       \renewcommand*\svgx@param@angle{0}%
1767       \FamilyKeyStateProcessed%
1768     }{%
1769       \Ifstr{#1}{inherit}{%
1770         \renewcommand*\svgx@param@angle{\svg@param@angle}%
1771         \FamilyKeyStateProcessed%
1772       }{%
1773         \Ifisdimension{#1\p@}{%
1774           \renewcommand*\svgx@param@angle{#1}%
1775           \FamilyKeyStateProcessed%
1776         }{}%
1777       }%
1778     }%
1779   }%
1780 }
1781 </extract>

```

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

\svghidepreambleend

```

1782 <*base>
1783 \newcommand*\svghidepreamblestart{%
1784   \PackageWarning{svg}{%
1785     The macro '\string\svghidepreamblestart' is only meant\MessageBreak%
1786     to be used together with package 'svg-extract'.\MessageBreak%
1787     Nevertheless, nothing will happen%
1788   }%
1789 }
1790 \newcommand*\svghidepreambleend{%
1791   \PackageWarning{svg}{%
1792     The macro '\string\svghidepreambleend' is only meant\MessageBreak%
1793     to be used together with package 'svg-extract'.\MessageBreak%
1794     Nevertheless, nothing will happen%
1795   }%
1796 }
1797 </base>

```

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

```

1798 <*extract>
1799 \let\svghidepreamblestart\relax
1800 \let\svghidepreambleend\relax
1801 </extract>

```

### C.3. Auxiliary macros

\svg@extract  
 \svgx@stream@in  
 \svgx@read@line  
 \svgx@stream@out  
 \if@svgx@preamble@write

The macro \svg@extract does the actual job of both extracting and converting independent graphic files. Since it is necessary to run it with `--shell-escape` enabled, the command raises a warning if it is not activated. Afterwards, the package is finished.

```

1802 <*base>
1803 \newcommand*\svg@extract[1]{}
1804 </base>
1805 <*extract>
1806 \ifnum\pdf@shellescape=\@ne\relax\else%
1807   \renewcommand*\svg@extract[1]{%
1808     \if@svgx@run%
1809       \begingroup%

```

```

1810      \edef\svg@tempa{#1}%
1811      \svg@quotes@remove{\svg@tempa}%
1812      \PackageWarning{svg-extract}{%
1813        You didn't enable 'shell escape' (or 'write18')\MessageBreak%
1814        so it wasn't possible to run the extraction for\MessageBreak%
1815        file '\svg@tempa'\MessageBreak%
1816      }%
1817      \endgroup%
1818      \fi%
1819    }%
1820    \expandafter\endinput%
1821 \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.

```

1822 \newread\svgx@stream@in
1823 \newcommand*\svgx@read@line{}
1824 \newwrite\svgx@stream@out
1825 \newif\if@svgx@preamble@write
1826 \renewcommand*\svgx@extract[1]{%

```

If option `extract` is enabled...

```

1827   \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<sup>A</sup>T<sub>E</sub>X 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.

```

1828   \if@svgx@out@sec%
1829     \svgx@get@out@sec%
1830   \fi%
1831   \stepcounter{svgx@out@count}%
1832   \begingroup%
1833     \def\svg@tempa##1.##2\@nil{%
1834       \IfArgIsEmpty{##2}{\edef\svgx@preamble{##1.\svgx@latex@ext}}{}%
1835     }%
1836     \expandafter\svg@tempa\svgx@preamble.\@nil%
1837     \IfFileExists{\svg@file@path\svgx@preamble}{%
1838       \@svg@file@foundtrue%
1839     }{%
1840       \svgx@get@path[]{\svgx@preamble}{\svgx@out@path}%
1841       \def\svg@tempa####1.####2\@nil{%
1842         \edef\svgx@preamble{\svg@file@name.####2}%
1843       }%
1844       \expandafter\svg@tempa\svgx@preamble.\@nil%
1845     }%
1846     \edef\svg@tempa{%
1847       \endgroup%
1848       \if@svg@file@found%
1849         \ifx\svg@file@path\@empty%
1850           \def\noexpand\svgx@preamble{.\svgx@preamble}%
1851         \else%
1852           \def\noexpand\svgx@preamble{\svg@file@path\svgx@preamble}%
1853         \fi%
1854       \fi%
1855     }%
1856     \svg@tempa%
1857     \begingroup%
1858       \endlinechar=\m@ne%
1859       \IfFileExists{\svgx@preamble}{%
1860         \PackageInfo{svg-extract}{%

```

```

1861         The preamble file '\svgx@preamble'\MessageBreak%
1862         is used for the generation of the auxiliary file\MessageBreak%
1863         '\svgx@out@name.\svgx@latex@ext'%
1864     }%

```

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

```

1865         \catcode'\#=12\relax%
1866         \immediate\openout\svgx@stream@out=\svgx@out@name.\svgx@latex@ext%
1867         \immediate\openin\svgx@stream@in=\svgx@preamble%
1868         \@svg@tempswatrue%
1869         \@svgx@preamble@writetrue%
1870         \def\svgx@read@line{}%

```

The given preamble file is read line by line and written to the separate auxiliary L<sup>A</sup>T<sub>E</sub>X file `\svgx@out@name.\svgx@latex@ext` via the output stream.

```

1871         \@whiles\if@svg@tempswa\fi{%
1872             \immediate\read\svgx@stream@in to\svgx@read@line%
1873             \ifx\svgx@read@line\@empty%
1874                 \ifeof\svgx@stream@in\@svg@tempswafalse\fi%
1875             \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`

```

1876         \svgx@read@preamble@till{\svghidepreamblestart}{}%
1877         \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.

```

1878         \svgx@read@preamble@till{\svgx@endpreamble}{\@svg@tempswafalse}%
1879         \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.

```

1880         \if@svgx@classfound\else%
1881             \expandafter\svgx@documentclass%
1882             \svgx@read@line\documentclass\documentclass\@nil%
1883         \fi%

```

Writing out the—maybe manipulated—read in line.

```

1884         \ifx\svgx@read@line\@empty\else%
1885             \immediate\write\svgx@stream@out{%
1886                 \unexpanded\expandafter{\svgx@read@line}%
1887             }%
1888         \fi%
1889         \fi%
1890         \fi%
1891     }%
1892     \immediate\closein\svgx@stream@in%
1893     \immediate\closeout\svgx@stream@out%
1894     \catcode'\#=6\relax%

```

Once the separate auxiliary L<sup>A</sup>T<sub>E</sub>X 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.

```

1895         \immediate\openin\svgx@stream@in=\svgx@out@name.\svgx@latex@ext%
1896         \def\svg@tempa{}%
1897         \loop\unless\ifeof\svgx@stream@in%
1898             \readline\svgx@stream@in to\svgx@read@line%
1899             \ifx\svgx@read@line\@empty\else%

```

```

1900         \edef\svg@tempa{%
1901             \unexpanded\expandafter{\svg@tempa}%
1902             \unexpanded\expandafter{\svgx@read@line}^^J%
1903         }%
1904         \fi%
1905         \repeat%
1906         \immediate\closein\svgx@stream@in%
1907     }{%

```

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

```

1908         \svg@quotes@remove{\svgx@preamble}%
1909         \ifx\svgx@preamble\@empty\else%
1910             \PackageWarning{svg-extract}{%
1911                 The preamble file '\svgx@preamble'\MessageBreak%
1912                 does not exist%
1913             }%
1914         \fi%
1915         \def\svg@tempa{%
1916     }%

```

After the preamble was read in and stored in `\svg@tempa`, the separate auxiliary L<sup>A</sup>T<sub>E</sub>X file is written again. Some information are written right at the beginning of the file.

```

1917         \immediate\openout\svgx@stream@out=\svgx@out@name.\svgx@latex@ext%
1918         \immediate\write\svgx@stream@out{%
1919             \@percentchar\@percentchar\space This file was generated by package
1920             'svg-extract'^^J%
1921             \@percentchar\@percentchar\space from source '\jobname'^^J%
1922             \@percentchar\@percentchar\space It's intended to be compiled with
1923             '\svgx@latex@exe\ifx\svgx@latex@opt\@empty\else\space\svgx@latex@opt\fi'
1924         }%

```

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.

```

1925         \immediate\write\svgx@stream@out{%
1926             \string\AtBeginDocument{\@percentchar^^J%
1927                 \space\space\string\svgxsetpapersize\@percentchar^^J%
1928                 \ifxetex\else\ifpdf\else%
1929                     \space\space\string\AtBeginDvi{\string\special{%
1930                         papersize=\string\the\string\paperwidth,%
1931                         \string\the\string\paperheight%
1932                     }}\@percentchar^^J%
1933                 \fi\fi%
1934             }^^J%
1935             \string\PassOptionsToPackage{hidelinks}{hyperref}%
1936         }%

```

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

```

1937         \if@svgx@classfound\else%
1938             \immediate\write\svgx@stream@out{\string\documentclass{article}}%
1939         \fi%

```

And now the stored preamble.

```

1940         \ifx\svg@tempa\@empty\else%
1941             \immediate\write\svgx@stream@out{\unexpanded\expandafter{\svg@tempa}}%
1942         \fi%

```

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

```
1943 \immediate\write\svgx@stream@out{\string\usepackage{svg-extract}}%
```

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

```
1944 \def\svg@tempa##1{%
1945 \immediate\write\svgx@stream@out{\string\svgsetup{##1}}%
1946 }%
1947 \if@svg@ink@latex\else%
1948 \svg@tempa{inkscape@latex=false}%
1949 \fi%
1950 \ifdim\svgx@param@width>\z@\relax%
1951 \svg@tempa{width=\svgx@param@width}%
1952 \fi%
1953 \ifdim\svgx@param@height>\z@\relax%
1954 \svg@tempa{height=\svgx@param@height}%
1955 \fi%
1956 \if@svgx@param@distort%
1957 \svg@tempa{distort=true}%
1958 \fi%
1959 \ifdim\dimexpr\svgx@param@scale\p@\relax=\p@\relax\else%
1960 \svg@tempa{scale=\svgx@param@scale}%
1961 \fi%
1962 \def\svg@tempb{\svg@param@pretex}%
1963 \ifx\svgx@param@pretex\svg@tempb\relax%
1964 \let\svgx@param@pretex\svg@param@pretex%
1965 \fi%
1966 \ifx\svgx@param@pretex\relax\else%
1967 \svg@tempa{pretex=\unexpanded\expandafter{\svgx@param@pretex}}%
1968 \fi%
1969 \def\svg@tempb{\svg@param@apptex}%
1970 \ifx\svgx@param@apptex\svg@tempb\relax%
1971 \let\svgx@param@apptex\svg@param@apptex%
1972 \fi%
1973 \ifx\svgx@param@apptex\relax\else%
1974 \svg@tempa{apptex=\unexpanded\expandafter{\svgx@param@apptex}}%
1975 \fi%
```

Parameter `lastpage` is only considered for including PDF files with L<sup>A</sup>T<sub>E</sub>X support.

```
1976 \let\svg@tempa\@empty%
1977 \if@svg@ink@latex%
1978 \Ifstr{\svg@ink@format}{pdf}{%
1979 \ifnum\value{svg@param@lastpage}>\z@\relax%
1980 \edef\svg@tempa{lastpage=\the\value{svg@param@lastpage}}%
1981 \else%
1982 \ifnum\value{svg@param@lastpage}=\z@\relax%
1983 \def\svg@tempa{lastpage=true}%
1984 \else%
1985 \def\svg@tempa{lastpage=false}%
1986 \fi%
1987 \fi%
1988 }{}%
1989 \fi%
```

The rotation angle, if given.

```
1990 \ifdim\dimexpr\svgx@param@angle\p@\relax=\z@\relax\else%
1991 \edef\svg@tempa{%
1992 angle=\svgx@param@angle\ifx\svg@tempa\@empty\else,\svg@tempa\fi%
1993 }%
1994 \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.

```

1995      \ifx\svg@tempa\@empty%
1996      \def\svg@tempa{\string\svgxsetbox{#1}}%
1997      \else%
1998      \edef\svg@tempa{\noexpand\string\noexpand\svgxsetbox[\svg@tempa]{#1}}%
1999      \fi%
2000      \immediate\write\svgx@stream@out{\svg@tempa}%

```

Package **xr** is used to evaluate possible labels within the included *Inkscape* L<sup>A</sup>T<sub>E</sub>X file.

```

2001      \if@svg@ink@latex%
2002      \IfFileExists{xr.sty}{%
2003      \immediate\write\svgx@stream@out{%
2004      \string\usepackage{xr}^^J%
2005      \string\externaldocument{\jobname}^^J%
2006      }%
2007      }{}%
2008      \fi%
2009      \immediate\write\svgx@stream@out{%
2010      \string\begin{document}^^J%
2011      \string\pagestyle{empty}^^J%
2012      \string\svgxoutputbox\@percentchar^^J%
2013      \string\end{document}%
2014      }%
2015      \immediate\closeout\svgx@stream@out%
2016      \endgroup%

```

After creating the separate auxiliary L<sup>A</sup>T<sub>E</sub>X file, the actual extraction and conversion can be done.

```

2017      \Ifstr{\svgx@format\svgx@cnv@format}{}{%
2018      \PackageWarning{svg-extract}{%
2019      Both keys ‘extractformat’ and ‘convertformat’ are\MessageBreak%
2020      empty, so nothing to do so far%
2021      }%
2022      }{}%

```

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**.

```

2023      \svg@quotes@remove{\svgx@out@path}%
2024      \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.

```

2025      \edef\svg@tempb{%
2026      \noexpand\svg@shell@mkdir{\svgx@out@path}%
2027      }%
2028      \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%

```

First of all the separate auxiliary L<sup>A</sup>T<sub>E</sub>X file is compiled with the detected L<sup>A</sup>T<sub>E</sub>X processor (`\svgx@latex@exe`) as often as defined by counter option `extractruns`.

```

2029      \edef\svg@tempb{%
2030      \noexpand\PackageInfo{svg-extract}{%
2031      Running LaTeX (\svgx@latex@exe) for graphic extraction%
2032      \ifx\svgx@latex@opt\@empty\else%
2033      \MessageBreak with added options ‘\svgx@latex@opt’%
2034      \fi%
2035      }%

```



```

2036 }%
2037 \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2038 \edef\svg@tempb{%
2039   \noexpand\ShellEscape{%
2040     \svgx@latex@exe\space\svgx@latex@opt\space%
2041     "\svgx@out@name.\svgx@latex@ext"%
2042   }%
2043 }%
2044 \loop\ifnum\value{svgx@runs}>\z@ \relax%
2045   \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2046   \advance\c@svgx@runs\m@ne%
2047 \repeat%

```

All files requested with option `extractformat` are created with internal conversion tools supplied by most L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> distributions if necessary.

```

2048 \def\svg@tempa##1##2##3{%
2049   \edef\svg@tempb{%
2050     \noexpand\ShellEscape{%
2051       \@nameuse{svgx@##1@exe}\space\@nameuse{svgx@##1@opt}\space%
2052       "\svgx@out@name.##2"%
2053     }%
2054   }%
2055   \AfterReadingMainAux{\PackageInfo{svg-extract}{Running ##1}}%
2056   \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2057 }%
2058 \@svg@tempswafalse%
2059 \ifxetex\else\ifpdf\else%
2060   \@svg@tempswattrue%
2061 \fi\fi%
2062 \if@svg@tempswa%
2063   \svg@tempa{dvips}{dvi}{ps}%
2064   \svgx@ifinlist{eps}{\svgx@format}{\svg@tempa{pstoeps}{ps}{eps}}{}%
2065   \svgx@ifinlist{pdf}{\svgx@format}{\svg@tempa{pstopdf}{ps}{pdf}}{}%
2066 \else%
2067   \svgx@ifinlist{eps}{\svgx@format}{\svg@tempa{pdftops}{pdf}{eps}}{}%
2068   \svgx@ifinlist{ps}{\svgx@format}{\svg@tempa{pdftops}{pdf}{ps}}{}%
2069 \fi%

```

Now the desired conversion tool is invoked if requested.

```

2070 \if@svgx@cnv@run%

```

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

```

2071 \ifx\svgx@cnv@dpi\relax%
2072   \ifx\svgx@cnv@dpi@png\@undefined%
2073     \def\svgx@cnv@dpi@png{300}%
2074   \fi%
2075 \fi%

```

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

```

2076 \expandafter\svgx@cnv@get@informat\expandafter{\svgx@format}%

```

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

```

2077 \@for\svg@tempa:=\svgx@cnv@format\do{%
2078   \ifx\svg@tempa\@empty\else%
2079     \expandafter\svgx@ifinlist\expandafter{\svg@tempa}{\svgx@format}{%
2080       \PackageWarning{svg-extract}{%
2081         File type ‘\svg@tempa’ was specified for option\MessageBreak%
2082         ‘extractformat’ (\svgx@format) as well as for \MessageBreak%
2083         option ‘convertformat’ (\svgx@cnv@format) so the\MessageBreak%
2084         conversion won’t be done%
2085       }%

```

```

2086      }{%
2087      \edef\svg@tempb{%
2088      \noexpand\PackageInfo{svg-extract}{%
2089      Converting '\svgx@out@name.\svgx@cnv@informat'\MessageBreak%
2090      to '\svgx@out@name.\svg@tempa'%
2091      }%
2092      }%
2093      \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2094      \edef\svg@tempb{%
2095      \noexpand\ShellEscape{%
2096      \svgx@cnv@cmd{\svgx@out@name}{\svgx@cnv@informat}{\svg@tempa}%
2097      }%
2098      }%
2099      \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2100      }%
2101      \fi%
2102      }%
2103      \fi%

```

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

```

2104      \edef\svg@tempa{\svgx@format@if@svgx@cnv@run,\svgx@cnv@format\fi}%
2105      \@for\svg@tempb:=\svg@tempa\do{%
2106      \ifx\svg@tempb\@empty\else%
2107      \edef\svg@tempb{%
2108      \noexpand\svgx@move{\svgx@out@name}{\svg@tempb}{\svgx@out@path}%
2109      }%
2110      \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2111      \fi%
2112      }%

```

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

```

2113      \@for\svg@tempa:=\svgx@clean\do{%
2114      \expandafter\svgx@ifinlist\expandafter{\svg@tempa}{\svg@tempb}{}%
2115      \edef\svg@tempb{%
2116      \noexpand\IfFileExists{"\svgx@out@name".\svg@tempa}{%
2117      \noexpand\svg@shell@rm{\svgx@out@name.\svg@tempa}%
2118      }{}%
2119      }%
2120      \expandafter\AtEndDocument\expandafter{%
2121      \expandafter\AfterReadingMainAux\expandafter{\svg@tempb}%
2122      }%
2123      }%
2124      }%
2125      }%
2126      \fi%
2127      }
2128      </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`.

```

2129 \newcommand*\svgx@out@sec{unknown}
2130 \newcommand*\svgx@get@out@sec{%
2131 \begingroup%
2132 \def\svg@tempa{}%
2133 \@for\svg@tempb:={%
2134 part,chapter,section,subsection,subsubsection,paragraph,subparagraph%
2135 }\do{%
2136 \ifx\svg@tempb\@empty\else%
2137 \scr@ifundefinedorrelax{the\svg@tempb}{}%
2138 \ifnum\value{\svg@tempb}>\z@&relax%
2139 \edef\svg@tempa{\svg@tempb}%
2140 \fi%

```

```

2141     }%
2142     \fi%
2143 }%
2144 \edef\svg@tempb{%
2145     \endgroup%
2146     \ifx\svg@tempa\@empty\else%
2147         \def\noexpand\svgx@out@sec{\csname the\svg@tempa\endcsname}%
2148     \fi%
2149 }%
2150 \svg@tempb%
2151 }

```

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

```

2152 \newif\ifsvgx@classfound
2153 \newcommand*\svgx@documentclass{}
2154 \def\svgx@documentclass#1\documentclass#2\documentclass#3\@nil{%
2155     \IfArgIsEmpty{#2}{-}{\@svgx@classfoundtrue}%
2156 }

```

`\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
2157 \newcommand*\svgx@read@preamble@till[2]{%
2158     \svgx@read@preamble@skip#1\@nil{till}{#2}%
2159 }
2160 \newcommand*\svgx@read@preamble@from[2]{%
2161     \svgx@read@preamble@skip#1\@nil{from}{#2}%
2162 }

```

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

```

2163 \newcommand*\svgx@read@preamble@skip{}
2164 \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.

```

2165     \def\svg@tempa##1{%
2166         \def\svg@tempa####1##1####2##1####3\@nil{%
2167             \IfArgIsEmpty{####3}{-}{%

```

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

```

2168         \Ifstr{#2}{till}{%
2169             \IfArgIsEmpty{####1}{-}{%
2170                 \immediate\write\svgx@stream@out{####1}%
2171             }%
2172             \@svgx@preamble@writefalse%
2173         }%

```

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

```

2174         \Ifstr{#2}{from}{%
2175             \IfArgIsEmpty{####2}{-}{%
2176                 \def\svgx@read@line{}%
2177             }%
2178             \def\svgx@read@line{####2}%
2179         }%
2180         \@svgx@preamble@writetrue%
2181     }-}%
2182 }%

```

Additonal stuff which should be done.

```
2183      #3%
2184    }%
2185  }%
2186 }%
```

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

```
2187 \edef\svg@tempb{\expandafter\detokenize\expandafter{#1}}%
2188 \expandafter\svg@tempa\expandafter{\svg@tempb}%
```

Calling the created macro.

```
2189 \edef\svg@tempb{%
2190   \expandafter\detokenize\expandafter{\svgx@read@line}\svg@tempb\svg@tempb%
2191 }%
2192 \expandafter\svg@tempa\svg@tempb\@nil%
2193 }
```

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

```
2194 \newcommand*\svgx@cnv@informat{}
2195 \newcommand*\svgx@cnv@get@informat[1]{%
2196   \begingroup%
2197   \def\svg@tempa##1,##2\@nil{%
2198     \def\svg@tempa{##1}%
2199   }%
2200   \svg@tempa#1,\@nil%
2201   \edef\svg@tempa{%
2202     \endgroup%
2203     \def\noexpand\svgx@cnv@informat{\svg@tempa}%
2204   }%
2205   \svg@tempa%
```

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

```
2206 \ifx\svgx@cnv@informat\@empty%
2207   \renewcommand*\svgx@cnv@informat{pdf}%
2208 \ifxetex\else\ifpdf\else%
2209   \renewcommand*\svgx@cnv@informat{ps}%
2210 \fi\fi%
2211 \fi%
2212 }
```

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

```
2213 \ifx\svgx@magick@exe\@empty
2214   \ifwindows
2215     \renewcommand*\svgx@magick@exe{magick}
2216   \else
2217     \renewcommand*\svgx@magick@exe{convert}
2218   \fi
2219 \fi
2220 \newcommand*\svgx@magick@cmd[3]{%
2221   \svgx@magick@exe\space%
2222   \svgx@useformatkey{\svgx@cnv@dpi}{#3}{-density }%
2223   \svgx@useformatkey{\svgx@magick@set}{#3}{}%
2224   "#1.#2"\space%
2225   \svgx@useformatkey{\svgx@magick@opr}{#3}{}%
2226   "#1.#3"%
2227 }
```

```

2228 \ifx\svgx@gs@exe\@empty
2229 \ifwindows
2230 \renewcommand*\svgx@gs@exe{gswin64c}
2231 \else
2232 \renewcommand*\svgx@gs@exe{gs}
2233 \fi
2234 \fi
2235 \newcommand*\svgx@gs@cmd[3]{%
2236 \svgx@gs@exe\space-dSAFER -dPATCH -dNOPAUSE\space%
2237 \svgx@useformatkey{svgx@gs@device}{#3}{-sDEVICE=}%
2238 \svgx@useformatkey{svgx@cnv@dpi}{#3}{-r}%
2239 \svgx@useformatkey{svgx@gs@opt}{#3}{}%
2240 -sOutputFile="#1.#3"\space"#1.#2"%
2241 }

```

`\svgx@move` If the file doesn't exist

```

2242 \newcommand*\svgx@move[3]{%
2243 \begin{group}%
2244 \IfFileExists{"#1".#2}{%
2245 \svg@shell@move{#1.#2}{#3#1.#2}%
2246 }{%
2247 \edef\svg@tempa{#2}%
2248 \@svg@tempswafalse%
2249 \expandafter\svgx@ifinlist\expandafter{\svg@tempa}{\svgx@cnv@format}{%
2250 \@svg@tempswatrue%
2251 \def\svg@tempb{conversion}%
2252 }{%
2253 \expandafter\svgx@ifinlist\expandafter{\svg@tempa}{pdf,ps,eps}{%
2254 \@svg@tempswatrue%
2255 \def\svg@tempb{extraction}%
2256 }{}%
2257 }%
2258 \if@svg@tempswa%
2259 \edef\svg@tempb{%
2260 The graphic file \svg@tempb\space failed\MessageBreak%
2261 for '#1.#2'\MessageBreak%
2262 Troubleshooting: Please check the log file how\MessageBreak%
2263 the invocation of the extraction took place and\MessageBreak%
2264 try to execute it yourself in the terminal%
2265 }%
2266 \else%
2267 \def\svg@tempb{%
2268 The extraction to format '#2' failed\MessageBreak%
2269 for '#1.#2'\MessageBreak%
2270 Only file types 'pdf,ps,eps' are supported for\MessageBreak%
2271 key 'exportformat'%
2272 }%
2273 \fi%
2274 \PackageWarning{svg-extract}{\svg@tempb}%
2275 }%
2276 \end{group}%
2277 }

```

`\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.

```

2278 \newcommand*\svgx@ifinlist[2]{%
2279 \begin{group}%
2280 \def\svg@tempa##1,##2\@nil{%
2281 \IfArgIsEmpty{##2}{%
2282 \aftergroup\@secondoftwo%
2283 }{%
2284 \aftergroup\@firstoftwo%
2285 }%

```

```

2286     }%
2287     \expandafter\svg@tempa\expandafter,#2,#1,\@nil%
2288 \endgroup%
2289 }

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

2290 \newcommand*\svgx@onlywindows[1]{%
2291 \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  $\langle key \rangle = \{ \langle format \rangle = \langle value \rangle \}$ .

2292 \newcommand*\svgx@ifkeyandval[3]{%
2293 \def\svg@tempa##1=##2=##3\@nil{\Ifstr{##3}{=}{#2}{#3}}%
2294 \svg@tempa#1==\@nil%
2295 }

\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.

2296 \newcommand*\svgx@cnv@get@dpi[1]{%
2297 \begingroup%
2298 \def\svg@tempa##1dpi##2x##3dpi##4\@nil{%
2299 \edef\svg@tempa{##1}%

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

2300 \@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.

2301 \Ifnumber{##1}{%
2302 \IfArgIsEmpty{##3}{\@svg@tempswatrue}{%
2303 \Ifnumber{##3}{\edef\svg@tempa{##1x##3}}{%
2304 }%
2305 }{%
2306 \if@svg@tempswa\else%
2307 \expandafter\svg@tempb\svg@tempa xx\@nil%
2308 \fi%
2309 }%

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

2310 \def\svg@tempb##1x##2x##3\@nil{%
2311 \Ifstr{##3}{x}{%
2312 \@svg@tempswatrue%
2313 \IfArgIsEmpty{##1}{\@svg@tempswafalse}{%
2314 \Ifnumber{##1}{\@svg@tempswafalse}%
2315 }%
2316 \IfArgIsEmpty{##2}{\@svg@tempswafalse}{%
2317 \Ifnumber{##2}{\@svg@tempswafalse}%
2318 }%
2319 \if@svg@tempswa%
2320 \edef\svg@tempa{##1x##2}%
2321 \fi%
2322 }{%
2323 }%
2324 \IfArgIsEmpty{#1}{%
2325 \let\svg@tempa\@empty%

```

```

2326 }{%
2327   \lowercase{\svg@tempa#1dpi#1xdpi\@nil}%
2328   \if@svg@tempswa\else%
2329     \let\svg@tempa\relax%
2330   \fi%
2331 }%
2332 \edef\svg@tempb{%
2333   \endgroup%
2334   \ifx\svg@tempa\relax%
2335     \let\noexpand\svg@tempa\noexpand\relax%
2336   \else%
2337     \def\noexpand\svg@tempa{\svg@tempa}%
2338   \fi%
2339 }%
2340 \svg@tempb%
2341 }

```

`\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.

```

2342 \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`.

```

2343   \svgx@ifkeyandval{#1}{%
2344     \svg@ifvalueisrelax{##2}{%
2345       \expandafter\let\csname #2@##1\endcsname\relax%
2346     }{%
2347       \@namedef{#2@##1}{##2}%
2348     }%

```

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

```

2349   }{%
2350     \svg@ifvalueisrelax{##1}{%
2351       \expandafter\let\csname #2\endcsname\relax%
2352     }{%
2353       \@namedef{#2}{##1}%
2354     }%
2355   }%
2356 }

```

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.

```

2357 \newcommand*\svgx@useformatkey[3]{%
2358   \scr@ifundefinedorrelax{#1@#2}{%
2359     \scr@ifundefinedorrelax{#1}{%
2360       \expandafter\ifx\csname #1\endcsname\@empty\else%
2361         #3\@nameuse{#1}\space%
2362       \fi%
2363     }%
2364     \scr@ifundefinedorrelax{#1@#2+}{%
2365       \expandafter\ifx\csname #1@#2+\endcsname\@empty\else%
2366         #3\@nameuse{#1@#2+}\space%
2367       \fi%
2368     }%
2369   }{%

```

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

```

2370 \expandafter\ifx\csname #1@#2\endcsname\@empty\else%
2371 #3\@nameuse{#1@#2}\space%
2372 \fi%
2373 }%
2374 }

```

## C.4. Commands for the separate auxiliary L<sup>A</sup>T<sub>E</sub>X-file

For the extraction of independent graphics, an auxiliary L<sup>A</sup>T<sub>E</sub>X file is needed. Within this file, the following commands are used to include the desired graphic.

<code>\svgxsetbox</code> <code>\svgx@setbox</code> <code>\if@svgx@standalone</code>	<p>Within the preamble of the auxiliary L<sup>A</sup>T<sub>E</sub>X 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. The macro <code>\svgx@setbox</code> is executed twice, the first time in the preamble and the second time at the very end of <code>\AtBeginDocument</code> if package <b>etoolbox</b> was loaded.</p>
---	--

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

```

2375 \newif\if@svgx@standalone
2376 \newcommand*\svgxsetbox[2] [] {%
2377 \@svgx@standalonetrue%
2378 \svgx@setbox{#1}{#2}%
2379 \scr@ifundefinedorrelax{AtEndPreamble}{%
2380 \let\svg@tempa\@firstofone%
2381 }{%
2382 \def\svg@tempa{\AtEndPreamble}%
2383 }%
2384 \svg@tempa{\AtBeginDocument{\svgx@setbox{#1}{#2}}}%
2385 }
2386 \newcommand*\svgx@setbox[2] {%
2387 \sbox\svg@box{\svg@@input[{#1},draft=false]{#2}}%
2388 \svgxsetpapersize%
2389 }

```

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

```

2390 \newcommand*\svgxsetpapersize{%
2391 \setlength\paperwidth{\the\wd\svg@box}%

```

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

```

2392 \scr@ifundefinedorrelax{stockwidth}{}%
2393 \setlength\stockwidth{\paperwidth}%
2394 }%
2395 \scr@ifundefinedorrelax{mediawidth}{}%
2396 \setlength\mediawidth{\paperwidth}%
2397 }%
2398 \setlength\textwidth{\paperwidth}%
2399 \setlength\paperheight{\the\dimexpr\ht\svg@box+\dp\svg@box\relax}%
2400 \scr@ifundefinedorrelax{stockheight}{}%
2401 \setlength\stockheight{\paperheight}%
2402 }%
2403 \scr@ifundefinedorrelax{mediaheight}{}%
2404 \setlength\mediaheight{\paperheight}%
2405 }%
2406 \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.

```

2407 \hoffset=-1in%
2408 \oddsidemargin=\z@%
2409 \evensidemargin=\z@%
2410 \voffset=-1in%
2411 \topmargin=\z@%
2412 \headheight=\z@%
2413 \headsep=\z@%
2414 \topskip=\z@%
2415 \footskip=\z@%
2416 \marginparsep=\z@%
2417 \marginparwidth=\z@%
2418 \marginparpush=\z@%
2419 }
2420 \@onlypreamble\svgxsetpapersize

```

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

```

2421 \newif\if@svgx@beamer
2422 \@ifclassloaded{beamer}{\@svgx@beamertrue}{}%
2423 \newcommand*\svgxoutputbox{%
2424   \begingroup%
2425     \setlength\parindent{\z@}%
2426     \setlength\parskip{\z@}%
2427     \setlength\parfillskip{\z@}%
2428     \if@svgx@beamer%
2429       \setbeamertemplate{navigation symbols}{}%
2430       \begin{frame}[plain]%
2431         \usebox\svg@box%
2432       \end{frame}%
2433     \else%
2434       \usebox\svg@box%
2435     \fi%
2436   \endgraf%
2437 \endgroup%
2438 }

```

## D. Processing Options

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

```

2439 <*base>
2440 \FamilyExecuteOptions{SVG}{%
2441   inkscape=true,inkscapepath=basesubdir,
2442   inkscapelatex=true,inkscapearea=drawing,distort=false,%
2443   usecolor=true,usetransparent=true%
2444 }
2445 </base>
2446 <*extract>
2447 \FamilyExecuteOptions{SVG}{%
2448   extract=true,extractpath=basesubdir,%
2449   extractruns=2,extractname=namenumbered,extractdistort=false,%
2450   convert=magick,convert=false,%
2451   gsdevice={png=png16m},gsdevice={jpeg=jpeg},gsdevice={jpg=jpeg},%
2452   gsdevice={tif=tiff48nc},gsdevice={tiff=tiff48nc},%
2453   gsdevice={eps=eps2write},gsdevice={ps=ps2write}%
2454 }
2455 </extract>
2456 \FamilyProcessOptions{SVG}

```

## E. Macros for file access

Finally, platform dependent 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.

```
2457 \ifnum\pdf@shellescape=\@ne\relax\else%
2458   \expandafter\endinput%
2459 \fi
2460 \RequirePackage{ifplatform}[2010/10/22]
```

```
\svg@shell@mkdir The platform dependent commands for file access.
\svg@shell@@mkdir
  \svg@shell@mv    2461 \ifwindows
  \svg@shell@@mv   2462   \newcommand*\svg@shell@@mkdir[1]{if not exist "#1" mkdir "#1"}
  \svg@shell@rm    2463   \newcommand*\svg@shell@@mv{move}
  \svg@shell@@rm   2464   \newcommand*\svg@shell@@rm{del}
  \svg@shell@@rm   2465 \else
  \svg@shell@@rm   2466   \newcommand*\svg@shell@@mkdir[1]{mkdir -p "#1"}
  \svg@shell@@rm   2467   \newcommand*\svg@shell@@mv{mv}
  \svg@shell@@rm   2468   \newcommand*\svg@shell@@rm{rm}
  \svg@shell@@rm   2469 \fi
```

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

```
2470 \newcommand*\svg@shell@mkdir[1]{%
2471   \begingroup%
2472     \svg@quotes@remove[#{1}]{\svg@tempa}%
2473     \@svg@tempswattrue%
2474     \Ifstr{\svg@tempa}{.}{\@svg@tempswafalse}{%
2475       \Ifstr{\svg@tempa}{.}{\@svg@tempswafalse}{%
2476         }%
2477         \if@svg@tempswa%
2478           \ShellEscape{\svg@shell@@mkdir{\svg@tempa}}%
2479         \fi%
2480       \endgroup%
2481 }
```

Commands for moving and deleting files.

```
2482 \newcommand*\svg@shell@move[2]{%
2483   \ShellEscape{\svg@shell@@mv\space"#1"\space"#2"}%
2484 }
2485 \newcommand*\svg@shell@rm[1]{%
2486   \ShellEscape{\svg@shell@@rm\space"#1"}%
2487 }
```

At the very end, the catcodes are restored.

```
2488 \svg@catcodecodes@restore
```

# Index

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

<b>A</b>	
apptex (opt.)	5, <a href="#">309</a>
<b>C</b>	
clean (opt.)	9, <a href="#">1717</a>
clear (opt.)	<a href="#">1717</a>
convert (opt.)	10, <a href="#">1344</a>
convertdensity (opt.)	<a href="#">1447</a>
convertdpi (opt.)	10, <a href="#">1447</a>
convertformat (opt.)	10, <a href="#">1420</a>
counters:	
svg@param@lastpage	<a href="#">332</a>
svgx@out@count	<a href="#">1516</a>
svgx@runs	<a href="#">1240</a>
<b>D</b>	
distort (opt.)	5, <a href="#">252</a>
draft (opt.)	6, <a href="#">346</a>
dvipsopt (opt.)	9, <a href="#">1295</a>
<b>E</b>	
end (opt.)	<a href="#">1213</a>
eps (opt.)	<a href="#">1172</a>
exclude (opt.)	9, <a href="#">1738</a>
ext (opt.)	<a href="#">214</a>
extension (opt.)	<a href="#">214</a>
extract (opt.)	8, <a href="#">1140</a>
extractapptex (opt.)	8, <a href="#">1681</a>
extractdistort (opt.)	8, <a href="#">1585</a>
extractformat (opt.)	8, <a href="#">1172</a>
extractheight (opt.)	8, <a href="#">1585</a>
extractkeepaspectratio (opt.)	<a href="#">1585</a>
extractname (opt.)	8, <a href="#">1516</a>
extractpath (opt.)	8, <a href="#">1516</a>
extractpostex (opt.)	<a href="#">1681</a>
extractpreamble (opt.)	8, <a href="#">1213</a>
extractpreambleend (opt.)	8, <a href="#">1213</a>
extractpretex (opt.)	8, <a href="#">1681</a>
extractruns (opt.)	9, <a href="#">1240</a>
extractscale (opt.)	8, <a href="#">1585</a>
extractwidth (opt.)	8, <a href="#">1585</a>
<b>G</b>	
gsdevice (opt.)	11, <a href="#">1494</a>
gsexex (opt.)	11, <a href="#">1494</a>
gsopt (opt.)	11, <a href="#">1494</a>
<b>H</b>	
height (opt.)	5, <a href="#">252</a>
<b>I</b>	
\if@svg@draft	<a href="#">346</a>
\if@svg@file@found	<a href="#">487</a>
\if@svg@ink@run	<a href="#">847</a>
\if@svg@param@distort	<a href="#">252</a>
\if@svg@quotes@found	<a href="#">385</a>
\if@svg@tempswa	<a href="#">12</a>
\if@svg@use@transparent	<a href="#">30</a>
\if@svg@use@xcolor	<a href="#">30</a>
\if@svgx@beamer	<a href="#">2421</a>
\if@svgx@classfound	<a href="#">2152</a>
\if@svgx@cnv@run	<a href="#">1344</a>
\if@svgx@out@sec	<a href="#">1516</a>
\if@svgx@preamble@write	<a href="#">1802</a>
\if@svgx@run	<a href="#">1140</a>
\if@svgx@standalone	<a href="#">2375</a>
\includeinkscape	6–7, <a href="#">764</a> , <a href="#">1760</a>
angle (param.)	6, <a href="#">805</a>
apptex (param.)	6, <a href="#">805</a>
clean (param.)	7, <a href="#">1760</a>
convert (param.)	7, <a href="#">1760</a>
convertdpi (param.)	7, <a href="#">1760</a>
convertformat (param.)	7, <a href="#">1760</a>
distort (param.)	6, <a href="#">805</a>
draft (param.)	6, <a href="#">805</a>
exclude (param.)	7, <a href="#">1760</a>
extract (param.)	7, <a href="#">1760</a>
extractangle (param.)	7, <a href="#">1760</a>
extractapptex (param.)	7, <a href="#">1760</a>
extractdistort (param.)	7, <a href="#">1760</a>
extractformat (param.)	7, <a href="#">1760</a>
extractheight (param.)	7, <a href="#">1760</a>
extractpreamble (param.)	7, <a href="#">1760</a>
extractpretex (param.)	7, <a href="#">1760</a>
extractruns (param.)	7, <a href="#">1760</a>
extractscale (param.)	7, <a href="#">1760</a>
extractwidth (param.)	7, <a href="#">1760</a>
gsdevice (param.)	7, <a href="#">1760</a>
gsopt (param.)	7, <a href="#">1760</a>
height (param.)	6, <a href="#">805</a>
inkscapeformat (param.)	6, <a href="#">805</a>
inkscapelatex (param.)	6, <a href="#">805</a>
lastpage (param.)	6, <a href="#">805</a>
latexopt (param.)	7, <a href="#">1760</a>
magickoperator (param.)	7, <a href="#">1760</a>
magicksetting (param.)	7, <a href="#">1760</a>
origin (param.)	6, <a href="#">805</a>
pretex (param.)	6, <a href="#">805</a>
scale (param.)	6, <a href="#">805</a>
width (param.)	6, <a href="#">805</a>
\includesvg	6–7, <a href="#">713</a> , <a href="#">1760</a>
angle (param.)	6, <a href="#">748</a>
apptex (param.)	6, <a href="#">716</a>
clean (param.)	7, <a href="#">1760</a>
convert (param.)	7, <a href="#">1760</a>
convertdpi (param.)	7, <a href="#">1760</a>
convertformat (param.)	7, <a href="#">1760</a>
distort (param.)	6, <a href="#">716</a>
draft (param.)	6, <a href="#">716</a>
exclude (param.)	7, <a href="#">1760</a>
extract (param.)	7, <a href="#">1760</a>
extractangle (param.)	7, <a href="#">1760</a>
extractapptex (param.)	7, <a href="#">1760</a>
extractdistort (param.)	7, <a href="#">1760</a>
extractformat (param.)	7, <a href="#">1760</a>
extractheight (param.)	7, <a href="#">1760</a>
extractpreamble (param.)	7, <a href="#">1760</a>
extractpretex (param.)	7, <a href="#">1760</a>
extractruns (param.)	7, <a href="#">1760</a>
extractscale (param.)	7, <a href="#">1760</a>
extractwidth (param.)	7, <a href="#">1760</a>

gsdevice (param.)	7, <a href="#">1760</a>
gsopt (param.)	7, <a href="#">1760</a>
height (param.)	6, <a href="#">716</a>
inkscape (param.)	6, <a href="#">716</a>
inkscapearea (param.)	6, <a href="#">716</a>
inkscapedpi (param.)	6, <a href="#">716</a>
inkscapeformat (param.)	6, <a href="#">716</a>
inkscapelatex (param.)	6, <a href="#">716</a>
inkscapeopt (param.)	6, <a href="#">716</a>
lastpage (param.)	6, <a href="#">745</a>
latexopt (param.)	7, <a href="#">1760</a>
magickoperator (param.)	7, <a href="#">1760</a>
magicksetting (param.)	7, <a href="#">1760</a>
origin (param.)	6, <a href="#">748</a>
pretex (param.)	6, <a href="#">716</a>
scale (param.)	6, <a href="#">716</a>
svgextension (param.)	6, <a href="#">716</a>
width (param.)	6, <a href="#">716</a>
inkscape (opt.)	4, <a href="#">54</a>
inkscapearea (opt.)	5, <a href="#">161</a>
inkscapedensity (opt.)	<a href="#">175</a>
inkscapedpi (opt.)	5, <a href="#">175</a>
inkscapeexe (opt.)	5, <a href="#">192</a>
inkscapeformat (opt.)	5, <a href="#">136</a>
inkscapelatex (opt.)	5, <a href="#">157</a>
inkscapename (opt.)	5, <a href="#">221</a>
inkscapeopt (opt.)	5, <a href="#">192</a>
inkscapepath (opt.)	4, <a href="#">221</a>

## K

keepaspectratio (opt.)	<a href="#">252</a>
------------------------	---------------------

## L

lastpage (opt.)	6, <a href="#">332</a>
latex (opt.)	<a href="#">157</a>
latexexe (opt.)	9, <a href="#">1257</a>
latexext (opt.)	9, <a href="#">1257</a>
latexopt (opt.)	9, <a href="#">1257</a>

## M

magickexe (opt.)	11, <a href="#">1472</a>
magickoperator (opt.)	11, <a href="#">1472</a>
magicksetting (opt.)	11, <a href="#">1472</a>

## N

name (opt.)	<a href="#">1516</a>
nottransparent (opt.)	3, <a href="#">30</a>
noxcolor (opt.)	3, <a href="#">30</a>

## O

off (opt.)	7, <a href="#">134</a> , <a href="#">1168</a>
on (opt.)	7, <a href="#">134</a> , <a href="#">1168</a>
options:	
apptex	5, <a href="#">309</a>
clean	9, <a href="#">1717</a>
clear	<a href="#">1717</a>
convert	10, <a href="#">1344</a>
convertdensity	<a href="#">1447</a>
convertdpi	10, <a href="#">1447</a>
convertformat	10, <a href="#">1420</a>
distort	5, <a href="#">252</a>
draft	6, <a href="#">346</a>
dvipsopt	9, <a href="#">1295</a>
end	<a href="#">1213</a>
eps	<a href="#">1172</a>
exclude	9, <a href="#">1738</a>

ext	<a href="#">214</a>
extension	<a href="#">214</a>
extract	8, <a href="#">1140</a>
extractapptex	8, <a href="#">1681</a>
extractdistort	8, <a href="#">1585</a>
extractformat	8, <a href="#">1172</a>
extractheight	8, <a href="#">1585</a>
extractkeepaspectratio	<a href="#">1585</a>
extractname	8, <a href="#">1516</a>
extractpath	8, <a href="#">1516</a>
extractpostex	<a href="#">1681</a>
extractpreamble	8, <a href="#">1213</a>
extractpreambleend	8, <a href="#">1213</a>
extractpretex	8, <a href="#">1681</a>
extractruns	9, <a href="#">1240</a>
extractscale	8, <a href="#">1585</a>
extractwidth	8, <a href="#">1585</a>
gsdevice	11, <a href="#">1494</a>
gsexex	11, <a href="#">1494</a>
gsopt	11, <a href="#">1494</a>
height	5, <a href="#">252</a>
inkscape	4, <a href="#">54</a>
inkscapearea	5, <a href="#">161</a>
inkscapedensity	<a href="#">175</a>
inkscapedpi	5, <a href="#">175</a>
inkscapeexe	5, <a href="#">192</a>
inkscapeformat	5, <a href="#">136</a>
inkscapelatex	5, <a href="#">157</a>
inkscapename	5, <a href="#">221</a>
inkscapeopt	5, <a href="#">192</a>
inkscapepath	4, <a href="#">221</a>
keepaspectratio	<a href="#">252</a>
lastpage	6, <a href="#">332</a>
latex	<a href="#">157</a>
latexexe	9, <a href="#">1257</a>
latexext	9, <a href="#">1257</a>
latexopt	9, <a href="#">1257</a>
magickexe	11, <a href="#">1472</a>
magickoperator	11, <a href="#">1472</a>
magicksetting	11, <a href="#">1472</a>
name	<a href="#">1516</a>
nottransparent	3, <a href="#">30</a>
noxcolor	3, <a href="#">30</a>
off	7, <a href="#">134</a> , <a href="#">1168</a>
on	7, <a href="#">134</a> , <a href="#">1168</a>
path	<a href="#">1516</a>
pdf	<a href="#">1172</a>
pdflatex	<a href="#">1257</a>
pdfstoeps	9, <a href="#">1295</a>
pdftops	<a href="#">1295</a>
pdftopsopt	9, <a href="#">1295</a>
png	<a href="#">1420</a>
postex	<a href="#">309</a>
preamble	<a href="#">1213</a>
pretex	5, <a href="#">309</a>
pstoeps	9, <a href="#">1295</a>
pstopdfopt	9, <a href="#">1295</a>
scale	5, <a href="#">252</a>
svgextension	5, <a href="#">214</a>
svgpath	<a href="#">202</a>
tex	<a href="#">157</a>
usetransparent	3, <a href="#">30</a>
usexcolor	3, <a href="#">30</a>
width	5, <a href="#">252</a>

## P

parameters:	
angle- <a href="#">\includeinkscape</a> . . . . .	<a href="#">6, 805</a>
angle- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 748</a>
apptex- <a href="#">\includeinkscape</a> . . . . .	<a href="#">6, 805</a>
apptex- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
clean- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
clean- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
convert- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
convert- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
convertdpi- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
convertdpi- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
convertformat- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
convertformat- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
distort- <a href="#">\includeinkscape</a> . . . . .	<a href="#">6, 805</a>
distort- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
draft- <a href="#">\includeinkscape</a> . . . . .	<a href="#">6, 805</a>
draft- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
exclude- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
exclude- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
extract- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
extract- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
extractangle- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
extractangle- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
extractapptex- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
extractapptex- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
extractdistort- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
extractdistort- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
extractformat- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
extractformat- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
extractheight- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
extractheight- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
extractpreamble- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
extractpreamble- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
extractpretex- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
extractpretex- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
extractruns- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
extractruns- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
extractscale- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
extractscale- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
extractwidth- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
extractwidth- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
gsdevice- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
gsdevice- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
gsop- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
gsop- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
height- <a href="#">\includeinkscape</a> . . . . .	<a href="#">6, 805</a>
height- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
inkscape- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
inkscapearea- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
inkscapedpi- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
inkscapeformat- <a href="#">\includeinkscape</a> . . . . .	<a href="#">6, 805</a>
inkscapeformat- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
inkscapelatex- <a href="#">\includeinkscape</a> . . . . .	<a href="#">6, 805</a>
inkscapelatex- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
inkscapeopt- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
lastpage- <a href="#">\includeinkscape</a> . . . . .	<a href="#">6, 805</a>
lastpage- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 745</a>
latexopt- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
latexopt- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
magickoperator- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>
magickoperator- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
magicksetting- <a href="#">\includeinkscape</a> . . . . .	<a href="#">7, 1760</a>

magicksetting- <a href="#">\includesvg</a> . . . . .	<a href="#">7, 1760</a>
origin- <a href="#">\includeinkscape</a> . . . . .	<a href="#">6, 805</a>
origin- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 748</a>
pretex- <a href="#">\includeinkscape</a> . . . . .	<a href="#">6, 805</a>
pretex- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
scale- <a href="#">\includeinkscape</a> . . . . .	<a href="#">6, 805</a>
scale- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
svgextension- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>
width- <a href="#">\includeinkscape</a> . . . . .	<a href="#">6, 805</a>
width- <a href="#">\includesvg</a> . . . . .	<a href="#">6, 716</a>

path (opt.) . . . . .	<a href="#">1516</a>
pdf (opt.) . . . . .	<a href="#">1172</a>
pdflatex (opt.) . . . . .	<a href="#">1257</a>
pdftoept (opt.) . . . . .	<a href="#">9, 1295</a>
pdftops (opt.) . . . . .	<a href="#">1295</a>
pdftops (opt.) . . . . .	<a href="#">9, 1295</a>
png (opt.) . . . . .	<a href="#">1420</a>
postex (opt.) . . . . .	<a href="#">309</a>
preamble (opt.) . . . . .	<a href="#">1213</a>
pretex (opt.) . . . . .	<a href="#">5, 309</a>
pstoept (opt.) . . . . .	<a href="#">9, 1295</a>
pstopdft (opt.) . . . . .	<a href="#">9, 1295</a>

## S

scale (opt.) . . . . .	<a href="#">5, 252</a>
<a href="#">\setsvg</a> . . . . .	<a href="#">699</a>
<a href="#">\svg@append@input@path</a> . . . . .	<a href="#">409</a>
<a href="#">\svg@box</a> . . . . .	<a href="#">12</a>
<a href="#">\svg@deactivate@dq</a> . . . . .	<a href="#">349</a>
<a href="#">\svg@deprecated@key</a> . . . . .	<a href="#">22</a>
<a href="#">\svg@deprecated@param</a> . . . . .	<a href="#">825</a>
<a href="#">\svg@dumy@key</a> . . . . .	<a href="#">1113</a>
<a href="#">\svg@extension@parse</a> . . . . .	<a href="#">564</a>
<a href="#">\svg@extension@@parse</a> . . . . .	<a href="#">564</a>
<a href="#">\svg@extract</a> . . . . .	<a href="#">1802</a>
<a href="#">\svg@file@base</a> . . . . .	<a href="#">487</a>
<a href="#">\svg@file@ext</a> . . . . .	<a href="#">214</a>
<a href="#">\svg@file@missing</a> . . . . .	<a href="#">600</a>
<a href="#">\svg@file@name</a> . . . . .	<a href="#">487</a>
<a href="#">\svg@file@path</a> . . . . .	<a href="#">487</a>
<a href="#">\svg@file@suffix</a> . . . . .	<a href="#">487</a>
<a href="#">\svg@filename@parse</a> . . . . .	<a href="#">527</a>
<a href="#">\svg@get@lastpage</a> . . . . .	<a href="#">930</a>
<a href="#">\svg@get@path</a> . . . . .	<a href="#">487</a>
<a href="#">\svg@ifilenewer</a> . . . . .	<a href="#">646</a>
<a href="#">\svg@ifvalueisrelax</a> . . . . .	<a href="#">476</a>
<a href="#">\svg@includegraphics@file</a> . . . . .	<a href="#">1095</a>
<a href="#">\svg@includegraphics@patched</a> . . . . .	<a href="#">1095</a>
<a href="#">\svg@includegraphics@saved</a> . . . . .	<a href="#">1063</a>
<a href="#">\svg@ink@area</a> . . . . .	<a href="#">161</a>
<a href="#">\svg@ink@cmd</a> . . . . .	<a href="#">922</a>
<a href="#">\svg@ink@dpi</a> . . . . .	<a href="#">175</a>
<a href="#">\svg@ink@exe</a> . . . . .	<a href="#">192</a>
<a href="#">\svg@ink@format</a> . . . . .	<a href="#">136</a>
<a href="#">\svg@ink@latex</a> . . . . .	<a href="#">157</a>
<a href="#">\svg@ink@mode</a> . . . . .	<a href="#">54</a>
<a href="#">\svg@ink@opt</a> . . . . .	<a href="#">192</a>
<a href="#">\svg@ink@run</a> . . . . .	<a href="#">847</a>
<a href="#">\svg@input</a> . . . . .	<a href="#">981</a>
<a href="#">\svg@input@path</a> . . . . .	<a href="#">701</a>
<a href="#">\svg@@input</a> . . . . .	<a href="#">981</a>
<a href="#">\svg@local@param@def</a> . . . . .	<a href="#">678</a>
<a href="#">\svg@local@param@set</a> . . . . .	<a href="#">678</a>
<a href="#">\svg@local@param@use</a> . . . . .	<a href="#">678</a>
<a href="#">\svg@normalize@path</a> . . . . .	<a href="#">450</a>
<a href="#">\svg@normalize@@path</a> . . . . .	<a href="#">450</a>

<code>\svg@out@base</code>	<a href="#">221</a>	<code>\svgx@ifkeyandval</code>	<a href="#">2292</a>
<code>\svg@out@name</code>	<a href="#">221</a>	<code>\svgx@latex@exe</code>	<a href="#">1257</a>
<code>\svg@out@path</code>	<a href="#">221</a>	<code>\svgx@latex@ext</code>	<a href="#">1257</a>
<code>\svg@param@apptex</code>	<a href="#">309</a>	<code>\svgx@latex@opt</code>	<a href="#">1257</a>
<code>svg@param@lastpage</code> (counter)	<a href="#">332</a>	<code>\svgx@magick@cmd</code>	<a href="#">2213</a>
<code>\svg@param@pretex</code>	<a href="#">309</a>	<code>\svgx@magick@exe</code>	<a href="#">1472</a>
<code>\svg@param@scale</code>	<a href="#">252</a>	<code>\svgx@magick@opr</code>	<a href="#">1472</a>
<code>\svg@param@width</code>	<a href="#">252</a>	<code>\svgx@magick@set</code>	<a href="#">1472</a>
<code>\svg@patches</code>	<a href="#">1063</a>	<code>\svgx@move</code>	<a href="#">2242</a>
<code>\svg@pictur@patched</code>	<a href="#">1072</a>	<code>\svgx@onlywindows</code>	<a href="#">2290</a>
<code>\svg@picture@saved</code>	<a href="#">1063</a>	<code>svgx@out@count</code> (counter)	<a href="#">1516</a>
<code>\svg@quotes@check</code>	<a href="#">385</a>	<code>\svgx@out@name</code>	<a href="#">1516</a>
<code>\svg@quotes@@check</code>	<a href="#">385</a>	<code>\svgx@out@path</code>	<a href="#">1516</a>
<code>\svg@quotes@remove</code>	<a href="#">360</a>	<code>\svgx@out@sec</code>	<a href="#">2129</a>
<code>\svg@quotes@@remove</code>	<a href="#">360</a>	<code>\svgx@param@apptex</code>	<a href="#">1681</a>
<code>\svg@remove@leadingchar</code>	<a href="#">393</a>	<code>\svgx@param@distort</code>	<a href="#">1585</a>
<code>\svg@sanitize@dq</code>	<a href="#">354</a>	<code>\svgx@param@pretex</code>	<a href="#">1681</a>
<code>\svg@set@input@path</code>	<a href="#">409</a>	<code>\svgx@param@scale</code>	<a href="#">1585</a>
<code>\svg@shell@mkdir</code>	<a href="#">2461</a>	<code>\svgx@param@width</code>	<a href="#">1585</a>
<code>\svg@shell@@mkdir</code>	<a href="#">2461</a>	<code>\svgx@pdftoops@exe</code>	<a href="#">1295</a>
<code>\svg@shell@mv</code>	<a href="#">2461</a>	<code>\svgx@pdftoops@opt</code>	<a href="#">1295</a>
<code>\svg@shell@@mv</code>	<a href="#">2461</a>	<code>\svgx@pdftops@exe</code>	<a href="#">1295</a>
<code>\svg@shell@rm</code>	<a href="#">2461</a>	<code>\svgx@pdftops@opt</code>	<a href="#">1295</a>
<code>\svg@shell@@rm</code>	<a href="#">2461</a>	<code>\svgx@preamble</code>	<a href="#">1213</a>
<code>\svg@tempa</code>	<a href="#">12</a>	<code>\svgx@pstoeps@exe</code>	<a href="#">1295</a>
<code>\svg@tempb</code>	<a href="#">12</a>	<code>\svgx@pstoeps@opt</code>	<a href="#">1295</a>
<code>\svg@wrn@scale</code>	<a href="#">964</a>	<code>\svgx@pstopdf@exe</code>	<a href="#">1295</a>
<code>svgextension</code> (opt.)	<a href="#">5</a> , <a href="#">214</a>	<code>\svgx@pstopdf@opt</code>	<a href="#">1295</a>
<code>\svghidepreambleend</code>	<a href="#">9</a> , <a href="#">1782</a>	<code>\svgx@read@line</code>	<a href="#">1802</a>
<code>\svghidepreamblestart</code>	<a href="#">9</a> , <a href="#">1782</a>	<code>\svgx@read@preamble@from</code>	<a href="#">2157</a>
<code>\svgpath</code>	<a href="#">4</a> , <a href="#">701</a>	<code>\svgx@read@preamble@skip</code>	<a href="#">2157</a>
<code>svgpath</code> (opt.)	<a href="#">202</a>	<code>\svgx@read@preamble@till</code>	<a href="#">2157</a>
<code>\svgsetup</code>	<a href="#">3</a> , <a href="#">7</a> , <a href="#">699</a>	<code>svgx@runs</code> (counter)	<a href="#">1240</a>
<code>\svgx@clean</code>	<a href="#">1717</a>	<code>\svgx@setbox</code>	<a href="#">2375</a>
<code>\svgx@cnv@cmd</code>	<a href="#">1344</a>	<code>\svgx@setformatkey</code>	<a href="#">2342</a>
<code>\svgx@cnv@dpi</code>	<a href="#">1447</a>	<code>\svgx@stream@in</code>	<a href="#">1802</a>
<code>\svgx@cnv@format</code>	<a href="#">1420</a>	<code>\svgx@stream@out</code>	<a href="#">1802</a>
<code>\svgx@cnv@get@dpi</code>	<a href="#">2296</a>	<code>\svgx@useformatkey</code>	<a href="#">2342</a>
<code>\svgx@cnv@get@informat</code>	<a href="#">2194</a>	<code>\svgx@outputbox</code>	<a href="#">2421</a>
<code>\svgx@cnv@informat</code>	<a href="#">2194</a>	<code>\svgxsetbox</code>	<a href="#">2375</a>
<code>\svgx@documentclass</code>	<a href="#">2152</a>	<code>\svgxsetpapersize</code>	<a href="#">2390</a>
<code>\svgx@dvi@exe</code>	<a href="#">1295</a>		
<code>\svgx@dvi@opt</code>	<a href="#">1295</a>		
<code>\svgx@endpreamble</code>	<a href="#">1213</a>		
<code>\svgx@format</code>	<a href="#">1172</a>		
<code>\svgx@get@out@sec</code>	<a href="#">2129</a>		
<code>\svgx@gs@cmd</code>	<a href="#">2213</a>		
<code>\svgx@gs@device</code>	<a href="#">1494</a>		
<code>\svgx@gs@exe</code>	<a href="#">1494</a>		
<code>\svgx@gs@opt</code>	<a href="#">1494</a>		
<code>\svgx@ifinlist</code>	<a href="#">2278</a>		

## T

<code>tex</code> (opt.)	<a href="#">157</a>
-------------------------	---------------------

## U

<code>usetransparent</code> (opt.)	<a href="#">3</a> , <a href="#">30</a>
<code>usexcolor</code> (opt.)	<a href="#">3</a> , <a href="#">30</a>

## W

<code>width</code> (opt.)	<a href="#">5</a> , <a href="#">252</a>
---------------------------	---

## Change History

### v1.0

#### General

initial version by Philip Ilten	<a href="#">2</a>
---------------------------------	-------------------

### v2.00

#### General

new maintainer: Falk Hanisch	<a href="#">2</a>
package <b>subfig</b> not required anymore	<a href="#">2</a>
re-implementation from scratch	<a href="#">2</a>
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

<code>clean</code> (opt.): changes, file list possible	<a href="#">1717</a>
<code>convert</code> (opt.): changed/extended	<a href="#">1344</a>

convertdpi (opt.): new	1447
convertformat (opt.): new	1420
draft (opt.): new	346
dvipsopt (opt.): new	1295
end (opt.): deprecated	1213
eps (opt.): deprecated	1172
extract (opt.): new	1140
extractapptex (opt.): new	1681
extractformat (opt.): new	1172
extractheight (opt.): new	1585
extractname (opt.): new	1516
extractpath (opt.): new	1516
extractpreamble (opt.): new	1213
extractpreambleend (opt.): new	1213
extractpretex (opt.): new	1681
extractruns (opt.): new	1240
extractscale (opt.): new	1585
extractwidth (opt.): new	1585
gsdevice (opt.): new	1494
gsex (opt.): new	1494
gs (opt.): new	1494
height (opt.): new	252
\includeinkscape: new	764
\includesvg:	
changes, especially to optional	
parameters	713
angle (param.): new	748
draft (param.): new	716
height (param.): new	716
inkscape (param.): new	716
inkscapearea (param.): new	716
inkscapedpi (param.): new	716
inkscapeformat (param.): new	716
inkscapelatex (param.): new	716
inkscapeopt (param.): new	716
lastpage (param.): new	745
origin (param.): new	748
scale (param.): new	716
inkscape (opt.): changed/extended	54
inkscapearea (opt.): new	161
inkscapedpi (opt.): new	175
inkscapeexe (opt.): new	192
inkscapeformat (opt.): new	136
inkscapelatex (opt.): new	157
inkscapename (opt.): new	221
inkscapeopt (opt.): new	192
inkscapepath (opt.): new	221
lastpage (opt.): new	332
latexexe (opt.): new	1257
latexext (opt.): new	1257
latexopt (opt.): new	1257
magickexe (opt.): new	1472
magickoperator (opt.): new	1472
magicksetting (opt.): new	1472
name (opt.):	
deprecated	1516
support of <b>subfig</b> removed	1516
notransparent (opt.): new	30
noxc (opt.): new	30
off (opt.): new	134, 1168
on (opt.): new	134, 1168
path (opt.): deprecated	1516
pdf (opt.): deprecated	1172
pdflatex (opt.): deprecated	1257
pdf (opt.): deprecated	1295
pdftops (opt.): deprecated	1295

pdftops (opt.): new	1295
png (opt.): deprecated	1420
postex (opt.): deprecated	309
preamble (opt.): deprecated	1213
pstoeps (opt.): new	1295
pstopdf (opt.): new	1295
scale (opt.): new	252
\setsvg: deprecated	699
\svghidepreambleend: new	1782
\svghidepreamblestart: new	1782
\svgpath: new	701
svgpath (opt.): deprecated	202
\svgsetup: new	699
usetransparent (opt.): new	30
usexc (opt.): new	30

## v2.00a

### Implementation

\svgxsetpapersize: Bug fix for checking stock- and mediasizes	2390
---	------

## v2.00b

### Implementation

latex (opt.): new, alternative key for inkscapelatex	157
tex (opt.): new, alternative key for inkscapelatex	157

## v2.01

### General

new option <b>svgextension</b> to change the format of files exported by <b>Inkscape</b> from <b>svg</b> to a custom one	2
usage of <code>\input{&lt;tex filename&gt;}</code> within <b>Inkscape</b> graphics locates files in all declared searched folders	2

### Implementation

\includesvg:	
svgextension (param.): new	716
inkscape (opt.): using <code>\trim@spaces</code>	54
\svg@append@input@path: new	409
\svg@get@path:	
using <code>\svg@set@input@path</code>	487
using <code>\trim@spaces</code>	487
\svg@set@input@path: new	409
svgextension (opt.): new due to user request	214

## v2.02

### General

distortion of included and extracted graphics supported with options <b>distort</b> (or <b>keepaspectratio</b> ) and <b>extractdistort</b> as well as rotation for extractions ( <b>extractangle</b> )	2
fixed errors with active double quotes from <b>babel</b> in path arguments	2
multiple dots within file names possible	2
package <b>trimspaces</b> required	2

### Implementation

distort (opt.): new	252
extractdistort (opt.): new	1585
extractname (opt.): usage of <code>\svg@quotes@remove</code>	1516
extractpath (opt.): usage of <code>\svg@sanitize@dq</code>	1516
\if@svgx@standalone: new	2375



