The showexpl package*

Rolf Niepraschk (`Rolf.Niepraschk@gmx.de`)

2020/05/26

1 Introduction

The documentation of a LaTeX package is by far more readable if there are examples of the commands' and environments' usage. The best way to do that is to give a comparison of the LaTeX code and the formatted output. showexpl is a package for doing that comparison, it is based on the package listings which provides a good typesetted source code with emphasised keywords and so on.

2 Usage

You can use showexpl like every other package by putting the line

```
\usepackage{showexpl}
```

in your source code. showexpl doesn't know any options by itself, but all options for the underlying packages (listings and graphicx) will be passed to the respective packages.

showexpl provides one command and one environment:

- `\LTXinputExample` and
- LTXexample

`\LTXinputExample`

The syntax of `\LTXinputExample` is given by

`\LTXinputExample[`⟨*key val list*⟩`]{`⟨*file*⟩`}`

`LTXexample`

The syntax of the environment LTXexample is given by

`\begin{LTXexample}[`⟨*key val list*⟩`]`...`\end{LTXexample}`

The set of options represented by ⟨*key val list*⟩ is the same for both the command and the environment, the options are described in the following:

attachfile Boolean valued key, default value: false. If set to true the sourcecode will be attached to the .pdf file—presumed that the document is processed by `pdflatex`.

codefile Name of the (temporary) file that contains the code which will be formatted as source code. The default value is `\jobname.tmp`.

[*] This document corresponds to showexpl v0.3q, dated 2020/05/26.

explpreset A ⟨*key val list*⟩ which serves for presetting the properties of the formatting of the source code, for values see the documentation of the listings package. The default value is

graphic Name of a (graphic) file. This file—if present—will be included and displayed instead of the formatted code. The default value is empty.

hsep Defines the horizontal distance between the source code and the formatted text.

justification Defines the justification of the formatted text: reasonable values are `\raggedleft`, `\raggedright`, `\centering`. The default value is `\raggedright`.

overhang A *dimen*-value that defines the amount by which the formatted text and the source code can overlap the print space. The default value is 0 pt.

pos: Defines the relative position of the formatted text relating to the source code. Allowed values are `t`, `b`, `l`, `r`, `o`, and `i` for top, bottom, left, right, outer, and inner. The last values give sense only for two-sided printing, where there are outer and inner margins of a page. The default value is `l`.

preset Any TEX code executed before the sample code but not visible in the listings area.

rangeaccept Boolean valued key, default value is false. If set to true, one can define ranges of lines that will be excerpted from the source code.

rframe Defines the form of the frame around the formatted text. With a non-empty value (e. g. "single") a simple frame will be drawn. In the future more kinds of frames will be supported. The default value is empty (no frame).

varwidth Boolean valued key, default value is false. If set to true, the formatted text is set with its "natural" width instead of a fixed width as given by the value of the option `width`.

vsep Defines the vertical distance between the source code and the formatted text.

wide Boolean valued key, default value is false. If set to true, the source code and the formatted text overlap the print space and the margin area.

width A ⟨*dimen*⟩ value that defines the width of the formatted text. The default value depends of the relative positions of the source code and the formatted text.

scaled Without a value the formatted text will be scaled to fit the given width of the result area. With a number as value the formatted text will be scaled by this number.

In addition to these options the kind of the result box (default: `\fbox`) can be changed. For example:

```
\renewcommand\ResultBox{\fcolorbox{green}{lightgray}}
\setlength\ResultBoxSep{5mm}%  default: \fboxsep
\setlength\ResultBoxRule{2mm}% default: \fboxrule
```

3 Implementation

```
\DeclareOption{final}{%
  \PassOptionsToPackage{\CurrentOption}{graphicx}%
  \PassOptionsToPackage{\CurrentOption}{listings}%
}%
\DeclareOption{draft}{%
  \PassOptionsToPackage{\CurrentOption}{graphicx}%
  \PassOptionsToPackage{\CurrentOption}{listings}%
}%
```

```
\DeclareOption{attachfiles}{%
  \AtBeginDocument{\IfFileExists{attachfile.sty}%
    {\RequirePackage{attachfile}}{\def\SX@attachfile{}}}
}%
\DeclareOption*{\PassOptionsToPackage{\CurrentOption}{listings}}
\ProcessOptions\relax
\RequirePackage{refcount,listings,graphicx,varwidth,float}
```

We must aktivate code from package listings for writing files.

```
\lst@RequireAspects{writefile}
```

\SX@defaultWD Parameter #2 is a length or a number. Parameter #1 is a macro. After a call of \SX@defaultWD this macro contains the value of the length or the value of the number multiplied by \linewidth.

```
\newcommand*\SX@defaultWD[2]{%
  \afterassignment\SX@def@WD\dimen@#2\linewidth\relax{#1}}
\newcommand*\SX@def@WD{}
\def\SX@def@WD#1\relax#2{\edef#2{\the\dimen@}}
```

Additional keys.

```
\lst@Key{pos}\relax{\def\SX@pos{#1}}
\lst@Key{width}\relax{\def\SX@width{#1}}
\lst@Key{hsep}\relax{\@tempdima=#1\relax\edef\SX@hsep{\the\@tempdima}}
\lst@Key{vsep}\relax{\@tempdima=#1\relax\edef\SX@vsep{\the\@tempdima}}
\lst@Key{overhang}\relax{\def\SX@overhang{#1}}
\lst@Key{wide}f[t]{\lstKV@SetIf{#1}\if@SX@wide}
\lst@Key{rframe}\relax{\def\SX@rframe{#1}}
\lst@Key{preset}\relax{\def\SX@preset{#1}}
\newcommand*\SX@scaled{}
\lst@Key{scaled}{?}[!]{\def\SX@scaled{#1}}
```

```
\lst@Key{explpreset}\relax{\def\SX@explpreset{#1}}
\lst@Key{codefile}\relax{\def\SX@codefile{#1}}
\newif\if@SX@rangeaccept \@SX@rangeacceptfalse
\newif\if@SX@varwidth \@SX@varwidthfalse
\newif\if@SX@wide \@SX@widefalse
\newif\if@SX@attachfile \@SX@attachfilefalse
```

```
\lst@Key{rangeaccept}f[t]{\lstKV@SetIf{#1}\if@SX@rangeaccept}
```

```
\lst@Key{varwidth}f[t]{\lstKV@SetIf{#1}\if@SX@varwidth}
\lst@Key{justification}\relax{\def\SX@justification{#1}}
\lst@Key{attachfile}f[t]{\lstKV@SetIf{#1}\if@SX@attachfile}
\newcommand*\SX@graphicname{}%
\newcommand*\SX@graphicparam{}%
```

```
\lst@Key{graphic}{}[]{%
  \lstKV@OptArg[width=\linewidth]{#1}{%
    \edef\SX@graphicparam{##1}\edef\SX@graphicname{##2}%
  }%
}%
\newbox\SX@ResBox
\newcommand\ResultBox{} \let\ResultBox=\fbox
\newdimen\ResultBoxSep  \ResultBoxSep=\fboxsep
\newdimen\ResultBoxRule \ResultBoxRule=\fboxrule
\newcommand*\SX@pos{}
\newcommand*\SX@width{}
\newcommand*\SX@hsep{}
\newcommand*\SX@vsep{}
\newcommand*\SX@overhang{}
\newcommand*\SX@rframe{}
\newcommand\SX@preset{}
\newcommand*\SX@explpreset{}

\newcommand*\SX@@explpreset{}
\newcommand*\SX@codefile{}\edef\SX@codefile{\jobname.tmp}
\newcommand*\SX@justification{\raggedright}
```

`\SX@@preset` Contains some redefinitions of LaTeX macros and environments to do nothing. `\SX@@preset` will be called just before typesetting the result of the example code. More can be added with the user key "`preset=...`".

```
\newcommand*\SX@@preset{%
  \renewcommand\documentclass[2][]{\SX@eat@version}%
  \renewcommand\usepackage[2][]{\SX@eat@version}%
  \renewenvironment{document}{}{}%
  \renewcommand\cite[1][]{}%
  \let\tableofcontens\relax \let\listoffigures\relax
  \let\listoftables\relax \let\printindex\relax
  \let\listfiles\relax \let\nofiles\relax
  \let\index\@gobble \let\label\@gobble
  \let\refstepcounter=\@gobble
  \let\bibliography\@gobble
  \let\pagestyle\@gobble \let\thispagestyle\@gobble
  %%\let\immediate\relax \let\write\@gobbletwo
  %%\let\closeout\@gobble \let\@@input\@gobble
  \renewcommand\marginpar[2][]{}%
  \renewcommand\footnote[2][]{}%
  \let\@footnotetext\@gobble
  %%\abovedisplayskip=\z@
  %%\abovedisplayshortskip=\z@
}
\newcommand*\SX@eat@version[1][]{}
```

`\isSX@odd` Parameter `#1` is executed on odd pages, parameter `#2` on even pages.

```
\newif\ifSX@wasodd
\if@twoside
  \newcommand*\isSX@odd{%
    \begingroup
      \ifodd\getpagerefnumber{\SX@IDENT}%
        \aftergroup\SX@wasoddtrue
      \else
        \aftergroup\SX@wasoddfalse
      \fi
    \endgroup
    \ifSX@wasodd
      \expandafter\@firstoftwo
    \else
      \expandafter\@secondoftwo
    \fi
  }
\else
  \SX@wasoddtrue
  \newcommand*\isSX@odd[2]{#1}
\fi
```

The call of `\isSX@odd` sets also `\ifSX@wasodd` to true or false. If it's clear that no page break occurs, `\ifSX@wasodd` can be used.

```
\newcounter{ltxexample}
\newcommand*{\SX@IDENT}{SX@\number\value{ltxexample}}
```

`\SX@attachfile`

```
\newcommand*\SX@attachfile{%
  \if@SX@attachfile
    \attachfile[mimetype=text/plain,subject={example \theltxexample}]%
      {\SX@codefile}{}%
  \fi
}
```

`\SX@put@t/b/l/r/o/i` Six macros for positioning `#2` (result) and `#3` (code). The result can be above, below, left or right of the code area or on the outer or inner side. Parameter `#1` is the width of the result.

```
\newcommand*\SX@put@t[3]{%
  \SX@ResultArea{\linewidth}{#2}\endgraf\pagebreak[2]%
  \@tempdima=\dimexpr\SX@vsep\vskip\@tempdima
  \SX@CodeArea{\linewidth}{#3}%
}
\newcommand*\SX@put@b[3]{%
  \SX@CodeArea{\linewidth}{#3}\endgraf\pagebreak[2]%
  \@tempdima=\dimexpr\SX@vsep\vskip\@tempdima
  \SX@ResultArea{\linewidth}{#2}%
}
\newcommand*\SX@put@l[3]{%
  \@tempdimc=\dimexpr\linewidth-#1-\SX@hsep %
  \SX@ResultArea{#1}{#2}\hfill\SX@CodeArea{\@tempdimc}{#3}%
}
\newcommand*\SX@put@r[3]{%
  \@tempdimc=\dimexpr\linewidth-#1-\SX@hsep %
  \SX@CodeArea{\@tempdimc}{#3}\hfill\SX@ResultArea{#1}{#2}%
}
\newcommand*\SX@put@o[3]{%
  \@nameuse{SX@put@\ifSX@wasodd r\else l\fi}{#1}{#2}{#3}%
}
\newcommand*\SX@put@i[3]{%
  \@nameuse{SX@put@\ifSX@wasodd l\else r\fi}{#1}{#2}{#3}%
}
\newcommand\SX@ResultArea[2]{%
  \SX@justification\@tempdima=\dimexpr #1 %
  \parbox\@tempdima{#2}%
}
\newcommand\SX@CodeArea[2]{%
  \@tempdima=\dimexpr #1 %
  \sbox\@tempboxa{\parbox\@tempdima{#2}}%
  \@tempdima=\dp\@tempboxa\usebox\@tempboxa
  \rlap{\raisebox{-\@tempdima}[0pt][0pt]{\SX@attachfile}}%
}
\newcommand*\SX@KillAboveCaptionskip{%
  \ifx\lst@caption\@empty\else
    \lst@IfSubstring t\lst@captionpos
      {\vskip-\abovecaptionskip}{}%
  \fi
}
\newcommand*\SX@KillBelowCaptionskip{%
  \ifx\lst@caption\@empty\else
    \lst@IfSubstring b\lst@captionpos
      {\vskip-\belowcaptionskip}{}%
  \fi
}
```

LTXexample

```
\lstnewenvironment{LTXexample}[1][]{%
  \@temptokena{#1}%
  \begingroup
```

For ”codefile=...”/”graphic=...” if `\theltxexample` or `\thelstlisting` is part of the filename.

```
  \advance\c@ltxexample\@ne \advance\c@lstlisting\@ne
  \expandafter\lstset\expandafter{\SX@explpreset,#1}%
  \edef\x{\endgroup
    \def\noexpand\SX@codefile{\SX@codefile}%
    \def\noexpand\SX@graphicname{\SX@graphicname}%
    \def\noexpand\SX@graphicparam{\SX@graphicparam}}%
  \x
  \xdef\SX@@explpreset{\the\@temptokena,codefile=\SX@codefile,%
    graphic={[\SX@graphicparam]{\SX@graphicname}}}%
  \setbox\@tempboxa=\hbox\bgroup
  \lst@BeginWriteFile{\SX@codefile}%
}
{%
  \lst@EndWriteFile\egroup
  \SX@put@code@result
}
```

\SX@put@code@result

```
\newcommand*\SX@put@code@result{%
  \begingroup
    \expandafter\lstset\expandafter{\SX@explpreset}%
    \expandafter\lstset\expandafter{\SX@@explpreset}%
```

Use listings floating procedure if necessary.

```
  \ifx\lst@float\relax\else
    \edef\@tempa{\noexpand\lst@beginfloat{lstlisting}[\lst@float]}
    \expandafter\@tempa
  \fi
  \ifx\lst@caption\@empty
    \lstset{nolol=true}%
  \fi
  \if@SX@wide\def\SX@overhang{\marginparwidth+\marginparsep}\fi
  \trivlist\item\relax
    \stepcounter{ltxexample}\label{\SX@IDENT}%
```

Make `\SX@width` a real dimension if the unit is missing.

```
    \SX@defaultWD\SX@width{\SX@width}%
```

Set the default width if necessary.

```
    \ifdim\SX@width<\z@
      \@tempswatrue
      \def\@tempa{t}%
      \ifx\@tempa\SX@pos\@tempswafalse\fi
      \def\@tempa{b}%
      \ifx\@tempa\SX@pos\@tempswafalse\fi
      \@tempdima=\dimexpr\linewidth+\SX@overhang %
      \if@tempswa\@tempdima=.5\@tempdima\fi%
      \edef\SX@width{\the\@tempdima}%
    \fi
```

Correct `\SX@width` if a frame is requested.

```
    \ifx\SX@rframe\@empty
      \long\def\SX@frame##1{##1}%
    \else
      \let\SX@frame\ResultBox
      \@tempdima=\dimexpr\SX@width-2\ResultBoxSep-2\ResultBoxRule %
      \edef\SX@width{\the\@tempdima}%
    \fi
    \isSX@odd{\def\@tempa{l}}{\def\@tempa{r}}%
    \makebox[\linewidth][\@tempa]{%
      \parbox{\dimexpr\linewidth+\SX@overhang}{%
```

`\SX@codefile` (`\jobname.tmp`) is not nessesary for the filelist.

```
        \let\@addtofilelist\@gobble
        \let\lst@ifdisplaystyle=\iftrue
        \SX@KillAboveCaptionskip\lst@MakeCaption{t}%
```

Use the "natural" width of the result code if "varwidth" is true. .

```
        \setbox\SX@ResBox\hbox{%
          \fboxsep=\ResultBoxSep
          \fboxrule=\ResultBoxRule
          \SX@frame{%
            \@nameuse{\if@SX@varwidth varwidth\else minipage\fi}%
                \SX@width\relax
              \begingroup
                \SX@resultInput
              \endgroup
            \@nameuse{end\if@SX@varwidth varwidth\else minipage\fi}}}%
        \edef\SX@width{\the\wd\SX@ResBox}%
```

```
     \@ifundefined{SX@put@\SX@pos}%
      {\@latex@error{Parameter ‘\SX@pos’ undefined}\@ehd}%
     {\@nameuse{SX@put@\SX@pos}%
      {\SX@width}{\box\SX@ResBox}{\SX@codeInput}}%
     \lst@MakeCaption{b}\SX@KillBelowCaptionskip
    }%
   }%
  \endtrivlist
  \ifx\lst@float\relax\else\expandafter\lst@endfloat\fi
  \gdef\SX@@explpreset{}%
 \endgroup
}
```

```
\newcommand\SX@SkipToFirst{%
 \ifeof\@inputcheck\else
  \ifnum \lst@lineno=\lst@firstline\else
   \readline\@inputcheck to\SX@tempa
   \typeout{IGNORE (\the\lst@lineno)}%
   \global\advance\lst@lineno\@ne
   \SX@SkipToFirst
  \fi
 \fi
}
\newcommand\SX@ProcessResult{%
 \ifeof\@inputcheck
  \let\SX@tempb\relax
 \else
  \let\SX@tempb\SX@ProcessResult
  \ifnum \lst@lineno>\lst@lastline\relax
   \ifx\lst@linerange\@empty
    \let\SX@tempb\relax
   \else
    \lst@GetLineInterval
    \SX@SkipToFirst
   \fi
  \else
   \readline\@inputcheck to\SX@tempa
   \typeout{READ (\the\lst@lineno)}%
   \expandafter\g@addto@macro
    \expandafter\SX@lines\expandafter{\SX@tempa^^J}%
   \global\advance\lst@lineno\@ne
  \fi
 \fi
 \SX@tempb
}
```

\SX@input

```
\newcommand\SX@input[1]{%
 \begingroup
  \IfFileExists{#1}{}%
  {%
   \filename@parse{#1}%
   \ifx\filename@ext\relax \def\filename@ext{tex}\fi
   \@latexerr{File
```

```
        `\filename@area\filename@base.\filename@ext' not found.^^J^^J}\@ehd%
    }%
    \openin\@inputcheck#1
    \lsthk@PreSet\let\lst@linerange\@empty\global\lst@lineno\@ne
    \expandafter\lstset\expandafter{\SX@@explpreset}%
    \ifx\lst@linerange\@empty
      \edef\lst@linerange{{\lst@firstline}-{\lst@lastline},}%
    \fi
    \lst@GetLineInterval
    \SX@Info
    \newlinechar=`\^^J\relax
    \SX@SkipToFirst\let\SX@lines\@empty
    \SX@ProcessResult
    \closein\@inputcheck
    \scantokens\expandafter{\SX@lines}%
  \endgroup
}
```

```
\newcommand*\SX@Info{%
  \typeout{--------------------------------}%
  \typeout{pos=\SX@pos}%
  \typeout{width=\SX@width}%
  \typeout{hsep=\SX@hsep}%
  \typeout{vsep=\SX@vsep}%
  \typeout{overhang=\SX@overhang}%
  \typeout{rframe=\SX@rframe}%
  \typeout{codefile=\SX@codefile}%
  \@ifundefined{lst@firstline}{}%
    {\typeout{\string\lst@firstline=\lst@firstline}}%
  \@ifundefined{lst@lastline}{}%
    {\typeout{\string\lst@lastline=\lst@lastline}}%
  \@ifundefined{lst@linerange}{}%
    {\typeout{\string\lst@linerange=\lst@linerange}}%
  \typeout{\string\if@SX@wide=\if@SX@wide TRUE\else FALSE\fi}%
  \typeout{\string\if@SX@rangeaccept=\if@SX@rangeaccept TRUE\else FALSE\fi}%
  \typeout{\string\if@SX@varwidth=\if@SX@varwidth TRUE\else FALSE\fi}%
  \typeout{graphicfile=\SX@graphicname, graphicparameter=[\SX@graphicparam]}%
  \typeout{--------------------------------}%
}
\providecommand*\MakePercentIgnore{\catcode`\%9\relax}
\providecommand*\MakePercentComment{\catcode`\%14\relax}
```

\SX@resultInput

```
\newcommand*\SX@resultInput{%
  \ifx\SX@graphicname\@empty
    \begingroup
      \MakePercentComment\makeatother\catcode`\^^M=5\relax
      \SX@@preset\SX@preset
      \if@SX@rangeaccept
       \let\SX@tempa=\SX@input
      \else
       \let\SX@tempa=\input
      \fi
      \if\SX@scaled ?%
    \let\SX@tempb=\@firstofone
   \else
    \if\SX@scaled !%
     \def\SX@tempb##1{\resizebox{\SX@width}{!}{##1}}%
    \else
     \def\SX@tempb##1{\scalebox{\SX@scaled}{##1}}%
    \fi
   \fi
   \let\SX@lst@Init=\lst@Init
```

Prevents float environments from floating. This is not enough for floating listing environments! Why?

```
   \def\@xfloat##1[##2]{%
    \def\@captype{##1}%
    \@namedef{the\@captype}{0}%
    \@float@HH{##1}[H]}%
```

Special handling of floating listing environments.

```
   \def\lst@Init{%
    \let\lst@float=\relax
    \setcounter\@captype{-1}%
    \SX@lst@Init
   }
```

Typeset the Code.

```
   \SX@tempb{\SX@tempa{\SX@codefile}}\par
```

Restore the regular numbering of floats outside of ‘LTXexample’.

```
  \endgroup
 \else
  \expandafter\includegraphics\expandafter[\SX@graphicparam]%
   {\SX@graphicname}%
 \fi
}
```

\SX@codeInput

```
\newcommand*\SX@codeInput{%
```

Without a caption entry the command \lstinputlisting adds the filename to the “list of listings” (lol). This should be avoided.

```
 \begingroup
```

The default parameters for all examples.

```
 \expandafter\lstset\expandafter{\SX@explpreset}%
```

If ”numbers=none” then margin dimensions should be zero.

```
  \expandafter\lstset\expandafter{\SX@@explpreset}%
  \ifx\lst@PlaceNumber\@empty
   \g@addto@macro\SX@@explpreset{,xleftmargin=0pt,xrightmargin=0pt}%
  \fi
  \SX@Info
  \expandafter\lstinputlisting\expandafter[\SX@@explpreset,nolol=true,%
   caption={},belowskip=\z@,aboveskip=\z@,float=false]{\SX@codefile}%
 \endgroup
}%
```

```
\newcommand*\LTXinputExample[2][]{%
  \g@addto@macro\SX@@explpreset{float=false,#1,codefile=#2}%
  \SX@put@code@result}%
```

All the default values.

```
\lstset{explpreset={numbers=left,numberstyle=\tiny,numbersep=.3em,
```

Negative width means defaults.

```
  xleftmargin=1em,columns=flexible,language=[LaTeX]TEX},pos=l,width=-99pt,
  overhang=0pt,hsep=\columnsep,vsep=\bigskipamount,rframe=single}
```

.

```
\AtBeginDocument{%
  \def\theHlstnumber{\thelstlisting.\arabic{lstnumber}.\lst@neglisting}%
}
```

Changing the defaults possible in `showexpl.cfg`.

```
\InputIfFileExists{showexpl.cfg}{}{}
```

Change History

v0.1a
General: “hpos” and “vpos” added, “pos” removed (RN). . . 3
Initial version 1

v0.1b
`\SX@put@t/b/l/r/o/i`: Positioning the captions more independend of the result and code area (RN). 5

v0.1c
`\SX@put@t/b/l/r/o/i`: Commands `\SX@KillAboveCaptionskip` and `\SX@KillBelowCaptionskip` added (RN). 5

v0.1f
General: “lstpreset” added. (RN). 3

v0.1h
General: “codefile” added. (RN). . 3
“lstpreset” renamed to “explpreset” (RN). 3
New macro `\LTXinputExample` (RN). 11
Renamed from “example” to “LTXexample’ (RN). 6

v0.1i
General: Better caption positioning and correct distance between the parts (RN). 6

v0.1j
`\SX@input`: For ranges of lines (RN). 8
General: “rangeaccept” added (RN). 3

v0.1k
`\SX@put@t/b/l/r/o/i`: Change [a]bove to [t]op (RN). 5
General: Some bug corrections (RN). 3

v0.1l
General: “graphic” added (RN). . . 3

v0.1m
General: Problem related to `\label`/`\ref` solved (RN). 6

v0.2a
General: “varwidth” and “justification” added (RN). . . . 3
“varwidth” package used (RN). . 6

v0.2b
General: Check if `\SX@put@?` is defined (RN). 6

v0.3a
`\SX@attachfile`: Attach file functionality (with pdfTEX) added (RN). 5
General: “attachfile” added (RN). 3

v0.3b
`\SX@resultInput`: Input of result code now inside a group; `\makeatother` added (RN). . . . 9

v0.3c
`\SX@resultInput`: Wrong catcode for newline char corrected (RN). 9

v0.3d

`\SX@resultInput`: Missing `\par` added (RN). 9

v0.3e

`\SX@@preset`: More redefinitions added (RN). 4

v0.3g

General: `\SX@ProcessResult` is now working correctly using `\readline` and `\scantokens`. Thanks to Ulrich Diez for help (RN). 8

Missing `\newcommand` for `\SX@@explpreset` added (RN). 4

v0.3h

General: New Option ‘attachfiles’ (RN). 3

v0.3j

`\SX@put@code@result`: Setting `\lst@MakeCaption` to was a bad idea for hyperlinks. Group added to varwidth environment. (Suggestions by Ulrike Fischer.). 7

v0.3k

`\SX@put@code@result`: Setting `\lst@MakeCaption` to `\@gobble` again (prevent multiply defined labels; label key) 7

General: Definition for “hyperref” (suggested by Heiko Oberdiek) 11

v0.3l

`\SX@resultInput`: Code for “scaled” option (RN). 9

General: Option “scaled” and `\SX@scaled` added (RN). 3

v0.3m

`\SX@put@code@result`: Wrong assignement for `\lst@belowskip` (RN). 7

v0.3n

`\SX@put@code@result`: Use `\ResultBox` 7

General: Define `\ResultBox` etc. . . 3

Prevent utf8 encoding errors . . . 6

v0.3p

`\SX@@preset`: Remove extra treatment of ‘figure’/‘table’ (RN). 4

`\SX@put@code@result`: Let’s leave `\lst@MakeCaption` untouched (RN). 7

`\SX@resultInput`: Better handling of floats (RN). 10

`\isSX@odd`: Replace `\isodd` with `\ifodd\getpagerefnumber` (remove package ‘ifthen’) (RN). 4

General: Remove package ‘calc’ (RN). 3

v0.3q

`\SX@resultInput`: Floats should always be numbered 0 (RN). . 10

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; numbers in roman refer to the code lines where the entry is used.