The collectbox Package

Martin Scharrer
martin@scharrer-online.de

CTAN: http://www.ctan.org/pkg/collectbox

VC: https://bitbucket.org/martin_scharrer/collectbox

Version v0.4b – 2012/05/17

Abstract

This package provides macros to collect and process an macro argument (i.e. something which looks like a macro argument) as horizontal box instead as a real macro argument. These "arguments" will be stored like when using `\savebox`, `\sbox` or the `lrbox` environment and allow verbatim or other special code. Instead of explicit braces also implicit braces in the form of `\bgroup` and `\egroup` are supported. This allows to split the begin and end over different macros or to place them in the begin and end code of an environment. The provided macros are mainly intended to be used inside other macros or environments.

1 Quick overview

The following macros are provided for users (document authors) and LaTeX package authors/programmers and described in the following sections. The $*$ and [⟨...⟩] arguments are as always optional. The {⟨*box content*⟩} can also be written as `\bgroup`⟨*box content*⟩`\egroup` and be split across macro boundaries.

```
\collectbox*[⟨code at begin⟩]{⟨code⟩}[⟨code at end⟩]{⟨box content⟩}
\collectboxto{⟨box register⟩}{⟨code⟩}{⟨box content⟩}
\collectboxcheckenv{⟨name⟩}
```

```
\collectbox@{⟨code at begin⟩}{⟨code⟩}{⟨code at end⟩}{⟨box content⟩}
\@collectbox{⟨code⟩}{⟨box content⟩}
\@Collectbox{⟨code⟩}{⟨content⟩}
\@collectboxto{⟨box register⟩}{⟨token⟩}{⟨box content⟩}
\nocollectbox@{⟨code at begin⟩}{⟨code⟩}{⟨code at end⟩}
```

2 Dependencies, Compatibility and Installation

This package does not depend on any other LaTeX package or class. It should be compatible with all versions of LaTeX(DVI-LaTeX, pdfLaTeX, XeLaTeX and

LuaLaTeX). Colored content will be correctly handled.

This package should be (soon) part of the standard distributions TeXLive and MikTeX and can be installed over the provided package manager (i.e. with TeXLive: `tlmgr install collectcell`). This package can be also manually unpacked from `collectbox.dtx` by compiling the file `collectbox.ins` with LaTeX or TeX. This documentation can be created by compiling `collectbox.dtx` using pdfLaTeX in DVI or PDF mode. The unpackaged package file `collectbox.sty` should be copied to a newly created directory named `$TEXMF/tex/latex/collectbox/` under Linux or `%TEXMF%\tex\latex\collectbox\` under MS Windows where `$TEXMF` and `%TEXMF%` represents the local TeX tree. The documentation and README file can be copied to `$TEXMF/doc/latex/collectbox/`. Some TeX distributions require to update the list of files in the TeX tree, e.g. by running `texmf $TEXMF` afterwards. MikTeX users can use the graphical interface of the package manager.

3 User Interface

The following macros are provided on the user level.

3.1 Macros to collect boxes

`\collectbox{`⟨*code*⟩`}{`⟨*box content*⟩`}`
`\collectbox{`⟨*code*⟩`}\bgroup`⟨*box content*⟩`\egroup`

In its basic form this macro is written as `\collectbox{`⟨*code*⟩`}` and collects the following 'group' in explicit (`{ .. }`) or implicit (`\bgroup .. \egroup`) braces as box (here represented as `{`⟨*box content*⟩`}`). Afterwards the user provided ⟨*code*⟩ is executed. This code is processed inside an internal group and has access to the just collected content using `\BOXCONTENT` and other macros described further below. Usually the code does some calculations and/or modifications on the collected box and then typesets it using `\BOXCONTENT`.

An example is

```
\collectbox{\fbox{\BOXCONTENT}}{\verb+verbatim stuff \space\empty+}
```

which results in

verbatim stuff \space\empty

`\collectbox*{`⟨*code*⟩`}{`⟨*box content*⟩`}`
`\collectbox*{`⟨*code*⟩`}\bgroup`⟨*box content*⟩`\egroup`

Because very often the `\BOXCONTENT` is simply fed as argument to a macro at the end of the ⟨*code*⟩ a star version exists which adds `{\BOXCONTENT}` automatically to the end of the code.

An above example can be therefore simplified as:

```
\collectbox*{\fbox}{\verb+verbatim stuff \space\empty+}
```

which results in:

verbatim stuff \space\empty

Using this macro a **\fbox** variant can be defined which reads its content as real box and not as macro argument:

```
\newcommand{\Fbox}{\collectbox*{\fbox}}
```

\collectbox[⟨*code at begin*⟩]{⟨*code*⟩}[⟨*code at end*⟩]{⟨*box content*⟩}
\collectbox[⟨*code at begin*⟩]{⟨*code*⟩}[⟨*code at end*⟩]**\bgroup**⟨*box content*⟩**\egroup**
\collectbox*[⟨*code at begin*⟩]{⟨*code*⟩}[⟨*code at end*⟩]{⟨*box content*⟩}
\collectbox*[⟨*code at begin*⟩]{⟨*code*⟩}[⟨*code at end*⟩]**\bgroup**⟨*box content*⟩**\egroup**

Finally two optional arguments exists which allow the placement of further code at the beginning and end of the collected box. This code is part of the box and is expanded before the main ⟨*code*⟩ which is expanded after the box is fully collected. In other words **\collectbox**[⟨*code at begin*⟩]{⟨*code*⟩}[⟨*code at end*⟩]{⟨*box content*⟩} is basically the same as **\collectbox**{⟨*code*⟩}{⟨*code at begin*⟩⟨.⟩⟨*box content*⟩⟨.⟩⟨*code at end*⟩}, with the difference that the first form allows an user defined macro to set the begin and end code while the box content is provided by the user. Note that there is also some internal code ⟨.⟩ between ⟨*code at begin*⟩ and ⟨*box content*⟩ as well as between ⟨*box content*⟩ and ⟨*code at end*⟩.

Example: **\fbox** like macro which reads its "argument" as box and sets it green first:

```
\newcommand{\GFbox}{\collectbox*[\color{green}]{\fbox}}
\GFbox{test $a=4$ \verb|\relax|}
```

will result in: test $a = 4$ \relax

\collectedbox

This macro represents the box register defined by **\newsavebox** which holds the collected box. It can be used with the LaTeX's macro **\usebox**{⟨*box register*⟩} or with plainTeX macros like **\box**, **\copy**, **\unhbox** or **\unhcopy**.

\BOXCONTENT

This macro is short for **\usebox**{**\collectedbox**} and will place the collected box into the document. It can be used multiple times inside ⟨*code*⟩.

\width
\height
\depth
\totalheight

These macros represent the dimension of the collected box and can be used inside ⟨*code*⟩ . Here **\height** is the height of the box above the baseline and **\depth** the lengths how far the box is going below the baseline. Both values added together are provides as **\totalheight**. The box width is given by **\width**.

```
\collectboxto{⟨box register⟩}[⟨code at begin⟩]{⟨code⟩}[⟨code at end⟩]{⟨box content⟩}
\collectboxto{⟨box register⟩}[⟨code at begin⟩]{⟨code⟩}[⟨code at end⟩]\bgroup⟨box content⟩\egroup
```

This macro collects its last "argument" as horizontal box and stores into into the user provided ⟨*box register*⟩. The ⟨*code at begin*⟩ and ⟨*code at end*⟩ are executed at begin and end of the ⟨*box content*⟩ as part of the box. Afterwards the user provided ⟨*code*⟩ is executed. In contrast to `\collectbox` no group is added for both the box register assignment or the code and also none of the above auxiliary macros for the box dimensions and content can be used. The collected box can be typeset using `\usebox{`⟨*box register*⟩`}` and its width, height and depth dimensions can be accessed using the TEX primitives `\wd`⟨*box register*⟩, `\ht`⟨*box register*⟩, `\dp`⟨*box register*⟩.

Appropriate grouping must be added manually if the overall code should be kept local. Proper places to open and close the group is somewhere before the `\collectboxto` macro and in the ⟨*code*⟩ after the box was used, respectively. The box register must be declared beforehand using `\newsavebox` (LATEX) or `\newbox`(TEX). The provided box register `\collectedbox` can be used here but will be overwritten by further usages of most of the macros of this package. The LATEX code also provides `\@tempboxa` for temporary usages.

This macro can be used to create macros which collect more than one argument as box. For this two box registers are required and the ⟨*code*⟩ of the first `\collectboxto` usage must call it second time. The second ⟨*code*⟩ then typesets the two boxes.

3.2 Support for environments

```
\collectboxcheckenv{⟨name⟩}
```

This macro allows macros which use `\collectbox` to also be used as an environment. For this `\collectboxcheckenv` must be used before `\collectbox` with the macro name as argument. It then detects if the macro is used by itself or is called by `\begin{`⟨*name*⟩`}` and adjusts the internal configuration accordantly. In macro-mode it does not change anything, but in environment-mode a `\begingroup` is used to keep the configuration changes local. The corresponding `\end`⟨*name*⟩ macro need to close then both the box group using `\egroup` and then close the outer group `\engroup`. A suitable version is automatically defined if this macro doesn't exists yet. In environment-mode starting and trailing spaces are ignored which is also the normal behaviour of similar environments like `minipage`.

Example: Defines `\foobar` macro which can also be used as `foobar` environment.

```
\newcommand*{\foobar}{%
    \collectboxcheckenv{foobar}%
    \collectbox*{\fbox}%
}
```

4 Programmers Interface

For the package author/programmer and more advanced users the following internal macros are provided. If used inside a document file they must be wrapped inside **\makeatletter** and **\makeatother** (outside the macro which uses them).

\collectbox@{⟨*code at begin*⟩}{⟨*code*⟩}{⟨*code at end*⟩}{⟨*box content*⟩}

This macro is the internal form of **\collectbox** with the optional arguments replaced by mandatory ones. The **\collectbox** macro itself uses it after checking for and reading the optional arguments. If this macro is to be used in other macros this overhead can be avoided by using the internal form directly. Not used optional arguments can be simply kept empty. The star version is not supported but can be easily substituted by manually placing the **{\BOXCONTENT}** at the end of ⟨*code*⟩.

The above example can therefore be written in the faster processed form:

```
\makeatletter
\newcommand{\GFbox}{\collectbox@{\color{green}}{\fbox}{}}
\makeatother
\GFbox{test $a=4$ \relax}
```

which results in: test $a = 4$ \relax

\@collectbox{⟨*code*⟩}{⟨*box content*⟩}

This macro is a short version of **\collectbox** and only accepts one argument ⟨*code*⟩ besides the later ⟨*box content*⟩. It is intended for quick version for macros which do not need to insert code at begin or after the box content.

\@Collectbox{⟨*code*⟩}{⟨*content*⟩}

This macro is similar to **\@collectbox** but reads the content as macro argument and not as a box. This is more efficient but does not allow for special content like verbatim text. This macro is indented for applications when an already boxed but now modified content needs to be boxed again, like it is done by `adjustbox`.

\collectboxto@{⟨*code at begin*⟩}{⟨*code*⟩}{⟨*code at end*⟩}{⟨*box content*⟩}

This macro is the internal form of **\collectboxto** with the optional arguments replaced by mandatory ones. The **\collectboxto** macro itself uses it after checking for and reading the optional arguments. If this macro is to be used in other macros this overhead can be avoided by using the internal form directly. Not used optional arguments can be simply kept empty.

\@collectboxto{⟨*box register*⟩}{⟨*code*⟩}{⟨*box content*⟩}

This macro is a short version of **\collectboxto** and only accepts one ⟨*code*⟩ argument. It is intended for quick version for macros which do not need to insert

code at begin or after the box content.

`\nocollectbox@{`⟨*code at begin*⟩`}{`⟨*code*⟩`}{`⟨*code at end*⟩`}`

Turns a following brace group into basically `\hbox{`⟨*code at begin*⟩⟨*content*⟩⟨*code at end*⟩`}`⟨*code*⟩. This is useful if a box needs to be build like that but no actual processing of the content is required, e.g. for horizontal alignment only. Note that there are some internal code between the three code arguments.