The everyhook package*

Stephen Checkoway
`s@cs.jhu.edu`

November 26, 2014

Abstract

The everyhook package takes control of the six TEX token parameters `\everypar`, `\everymath`, `\everydisplay`, `\everyhbox`, `\everyvbox`, and `\everycr`. Real hooks for each of these can be installed using a stack like interface. For compatibility with LATEX standard classes and packages, each of the `\everyX` token lists can be set without interfering with the hooks.

Contents

1 Introduction

TEX contains nine token parameters, seven of which are inserted into the current list at various times. Quoting from *The TEXbook*, the seven token parameters of interest are[1]

`\everypar`	tokens to insert when a paragraph begins,
`\everymath`	tokens to insert when math in text begins,

---

[*] This document corresponds to everyhook v1.2, dated 2014/11/26.
[1] The remaining two token parameters are `\output` and `\errhelp`.

`\everydisplay` tokens to insert when display math begins,
`\everyhbox` tokens to insert when an hbox begins,
`\everyvbox` tokens to insert when a vbox begins,
`\everyjob` tokens to insert when the job begins, and
`\everycr` tokens to insert after every `\cr` or nonredundant `\crcr`.

Of these, `\everyjob` is not very useful outside of INITEX and so it won't be considered further.

The remaining six token parameters can be used to great effect. For example, the `\everypar` is used in `\paragraph` to set the title of the paragraph inline allowing constructions like

```
\paragraph{Paragraph title.}

A blank line followed by
the rest of the paragraph.
```

Paragraph title. A blank line followed by the rest of the paragraph.

which work properly rather than starting a new paragraph due to the blank line.

Similarly, `\everymath` and `\everydisplay` are used by the LaTeX kernel to set up math fonts.

Using the TeX primitives directly has the major downside that they cannot be used by multiple packages at the same time. Setting `\everypar` overwrites a prior usage. Even if one package is careful and always uses

```
\everypar=\expandafter{\the\everypar new tokens here}
```

so as not to stomp on another's usage, there's no guarantee that the other package will not later set **\everypar**={}.

To get around this, the everyhook package takes control of the six `\everyX` primitives listed above and for each one provides a stack like interface for two additional token lists, one to be expanded before the `\everyX` and one to be expanded after. For example,

```
\PushPreHook{hbox}{1}
\PushPreHook{hbox}{2}
\everyhbox={3}
\PushPostHook{hbox}{4}
\PushPostHook{hbox}{5}
```

will cause the insertion of the tokens `21345` at the start of an `\hbox`. Note that `\PushPreHook` adds tokens to the *left* of the list of tokens to appear before those in `\everyhbox` whereas `\PushPostHook` adds tokens to the *right* of the list of tokens to appear after those in `\everyhbox`.

2 Usage

The everyhook package has one (rather experimental) option, `excludeor` and is loaded using

```
\usepackage[excludeor]{everyhook}
```

or

```
\RequirePackage[excludeor]{everyhook}
```

as required where the option is, of course, optional.

2.1 Options

`excludeor`

Some of the hooks described below can cause unwanted behavior when active during the execution of LaTeX's output routine. The experimental `excludeor` option saves and clears the hooks at the beginning of the output routine and restores them at the end.

2.2 Manipulating hooks

There are 12 hooks, a pre and post hook for each of the six token parameters `par`, `math`, `display`, `hbox`, `vbox`, and `cr`. The first argument to all of the macros described in this section must be one of these six. **All hook manipulation is *global*.**

`\PushPreHook`
`\PopPreHook`

Pre hooks. Additional tokens ⟨*balanced text*⟩ are prepended to the pre hook ⟨*hook*⟩ using \PushPreHook{⟨*hook*⟩}{⟨*balanced text*⟩}. The most recently pushed tokens can be popped off using \PopPreHook{⟨*hook*⟩}.

`\PushPostHook`
`\PopPostHook`

Post hooks. Additional tokens ⟨*balanced text*⟩ are appended to the post hook ⟨*hook*⟩ using \PushPostHook{⟨*hook*⟩}{⟨*balanced text*⟩}. The most recently pushed tokens can be popped off using \PopPostHook{⟨*hook*⟩}.

`\SavePreHook`
`\SavePostHook`
`\RestorePreHook`
`\RestorePostHook`
`\ClearPreHook`
`\ClearPostHook`

Saving, restoring, and clearing hooks. Each of the 12 pre and post hooks can be saved to a macro, restored from a macro, or cleared independently. To save the pre hook ⟨*hook*⟩ to the macro `\cs`, use \SavePreHook{⟨*hook*⟩}{\cs}. Restoring is accomplished by \RestorePreHook{⟨*hook*⟩}{\cs}. To clear all of the tokens in a pre hook use \ClearPreHook{⟨*hook*⟩}. The `\SavePostHook`, `\RestorePostHook`, and `\ClearPostHook` are analogous.

3 Example

As a nontrivial example of where this package can be used, consider the following example.

```
\documentclass{article}
\usepackage{everyhook}
\usepackage{lipsum}

\begin{document}
\setlength{\parindent}{0pt}
\PushPreHook{par}{\llap{\textbullet\enskip}\null}
\paragraph{Lorem ipsum.}
\lipsum[1-4]
\PopPreHook{par}
\end{document}
```

This code will cause each paragraph of the *lorem ipsum* text to have no indentation and instead to place a bullet in the margin. See Figure 1. If `\everypar` were used instead, the `\paragraph` would replace the command to create the bullet with those needed to typeset the paragraph title.

Note that this package is not a panacea. We had to add a `\null` to the `par` hook because `\paragraph` uses `\lastbox` to remove the indentation box. Without the `\null` it ends up removing the box constructed by `\llap` instead.

Using the post `par` hook solves the `\lastbox` problem, but then the bullet is placed to the right of the `\paragraph` title.

Perhaps a better way to solve this problem is to remove the indentation box first, insert the bullet, and then place the box after. In this way, the bullet is always to the left of the paragraph indentation.

```
\PushPreHook{par}{{\setbox0=\lastbox
        \llap{\textbullet\enskip}\box0}}
```

4 Potential pitfalls

As noted in the previous section, it can be tricky to use the `par` hook correctly. This section contains an (almost certainly) incomplete list of pitfalls to watch out for when using `everyhook`.

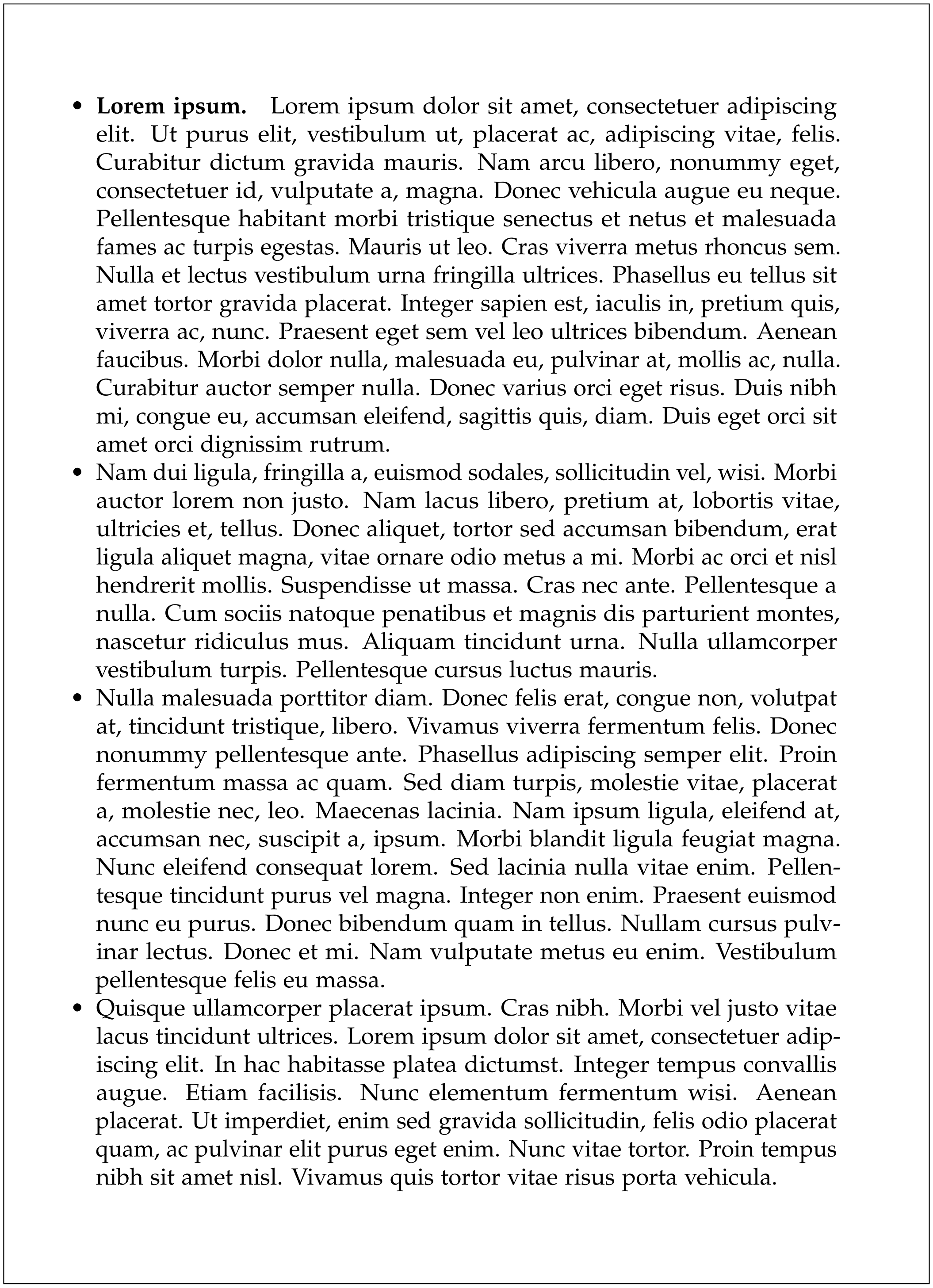
1. When using the `par` hooks, be aware that TeX will insert a box with the width of `\parindent` before the tokens in the pre hook. One way to handle this is to propogate the box to the right.

2. It is probably not a good idea to use the `hbox`, `vbox`, and `par` hooks at any place where TeX's output routine is likely to run. The `excludeor` option *should* help with this, but it might cause problems with other packages that also modify the output routine.

3. LaTeX's kernel takes control of the `\everymath` and `\everydisplay` token parameters to make its own adjustments in much the same way this package does. The `trace` package uses the kernel's private macros to insert its own hooks. It is probably best to only use the post `math` and `display` hooks to ensure that the kernel has done what it needs to do before you start typesetting stuff in math mode.

4. When using the `hbox` and `vbox` hooks, any hbox or vbox that appears in a `\setbox` will have the `\afterassignment` token inserted *before* the hooks. This is no different from TeX's normal behavior with `\afterassignment` and `\everyhbox`/`\everyvbox`, but can be surprising.

5. I'm sure there are others.



- **Lorem ipsum.** Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.
- Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.
- Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.
- Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Figure 1: Example output.

5 Implementation

The package begins with the usual package identification and then it declares the one option `excludeor` and finally it loads the `etoolbox` package. This package is not strictly necessary, but it does simplify some stuff and provides handy macros for dealing with control sequence names.

```
\NeedsTeXFormat{LaTeX2e}[1999/12/01]
\RequirePackage{svn-prov}
\ProvidesPackageSVN
        {$Id: everyhook.dtx 12 2014-11-26 15:34:56Z steve $}
        [v1.2 \revinfo\ Hooks for low level TeX everyX primitives.]
\DeclareOption{excludeor}{%
        \AtBeginDocument{%
                \output\expandafter{%
                        \expandafter\eh@saveclearallhooks
                        \the\output
                        \eh@restoreallhooks
                }%
        }%
}
\ProcessOptions\relax
\RequirePackage{etoolbox}
```

`\eh@definehook` This performs all of the setup work for each hook. First, it takes control of TeX's token parameter given in the second argument. Then it shadows the name of the primitive with a normal token register (and copies the current definition). The pre and post hooks are defined to be initially empty.

```
\def\eh@definehook#1#2{%
        \cslet{eh@every#1}#2%
        \newtoks#2%
        \cslet{eh@private#1}#2%
        #2\csuse{eh@every#1}%
        \csdef{eh@pre#1}{}%
        \csdef{eh@post#1}{}%
```

This is slightly tricky to get right. Basically, we want to set the `\everyfoo` primitive which we have saved as `\eh@everyfoo` like

`\eh@everyfoo={\eh@prefoo`**\the\expandafter**`\everyfoo\eh@postfoo}`.

The reason for the `\expandafter` is to make sure it is expanded before the the token register `\everyfoo` is expanded. Thus if the post hook is empty, then code in `\everyfoo` sees no additional tokens, in case that is important.

Unfortunately, some code wants to redefine `\everyfoo` itself in order to prevent *other* code that uses `\everyfoo` from actually setting anything. To deal with that, we use the private token list

`\eh@everyfoo{\eh@prefoo`**\the\expandafter**`\eh@privatefoo\eh@postfoo}`

```
        \csuse{eh@every#1}\expandafter{\csname eh@pre#1\expandafter\endcsname
                \expandafter\the\csname eh@private#1\expandafter\endcsname
                \csname eh@post#1\endcsname}%
}
```

`\everypar` Define the hooks for the par hook.

```
\eh@definehook{par}\everypar
```

`\frozen@everymath`
`\frozen@everydisplay`

Define the `math` and `display` hooks. Since the LaTeX kernel has already saved `\everymath` and `\everydisplay` into the frozen macros, we take control by redefining the frozen ones instead.

```
\eh@definehook{math}\frozen@everymath
\eh@definehook{display}\frozen@everydisplay
```

`\everyhbox`
`\everyvbox`
`\everycr`

Define the `hbox`, `vbox`, and `cr` hooks and free up some used memory.

```
\eh@definehook{hbox}\everyhbox
\eh@definehook{vbox}\everyvbox
\eh@definehook{cr}\everycr
\undef\eh@definehook
```

`\eh@hookseparator`

An separator used to separate tokens in each hook.

```
\def\eh@hookseparator{}
```

`\eh@checkhook`

Check that the hook is one of the six.

```
\def\eh@checkhook#1#2{%
        \ifcsdef{eh@every#1}{}{\PackageError{everyhook}{Argument #1 to
        \protect#2\space is invalid}{There is no hook for
        \protect\every#1.}}%
}
```

`\eh@checkhooknotempty`

Check that the hook is both defined and not empty so that we can pop.

```
\def\eh@checkhooknotempty#1#2#3{%
        \eh@checkhook{#2}#3%
        \ifcsempty{eh@#1#2}{\PackageError{everyhook}{The #1 hook for
        \protect\every#2\space is empty}{I have seen too many
        \protect#3{#2}s.}{}}%
}
```

`\PushPreHook`

Prepend tokens to the pre hook, separated via the separator.

```
\newrobustcmd\PushPreHook[2]{%
        \eh@checkhook{#1}\PushPreHook
        \def\eh@tempi{#2}%
        \letcs\eh@tempii{eh@pre#1}%
        \expandafter\gdef\csname eh@pre#1\expandafter\expandafter
                \expandafter\endcsname\expandafter\expandafter
                \expandafter{\expandafter\eh@tempi\expandafter
                                \eh@hookseparator\eh@tempii}%
        \undef\eh@tempi
        \undef\eh@tempii
}
```

`\PopPreHook`
`\eh@popprehook`

Check that the hook is not empty, and then pop off the left tokens and separator. We can use delimited parameters to strip off the first set of tokens.

```
\newrobustcmd\PopPreHook[1]{%
        \eh@checkhooknotempty{pre}{#1}\PopPreHook
        \expandafter\eh@popprehook\csname eh@pre#1\expandafter
                \expandafter\expandafter\endcsname
                \csname eh@pre#1\endcsname\eh@hookend
}
\def\eh@popprehook#1#2\eh@hookseparator#3\eh@hookend{\gdef#1{#3}}
```

`\PushPostHook`

Append a separator and tokens to the post hook.

```
\newrobustcmd\PushPostHook[2]{%
        \eh@checkhook{#1}\PushPostHook
        \letcs\eh@tempi{eh@post#1}%
        \expandafter\gdef\csname eh@post#1\expandafter\endcsname
                \expandafter{\eh@tempi\eh@hookseparator#2}%
        \undef\eh@tempi
}
```

`\PopPostHook`
`\eh@popposthook`
`\eh@sentinel`

Check that the post hook is not empty. Then, iterate over the tokens in the list until we reach the end and strip that off.

```
\newrobustcmd\PopPostHook[1]{%
        \eh@checkhooknotempty{post}{#1}\PopPostHook
        \letcs\eh@tempi{eh@post#1}%
        \expandafter\eh@popposthook\csname eh@post#1\expandafter
                \endcsname\expandafter{\expandafter}\eh@tempi
                \eh@hookend\eh@hookseparator\eh@sentinel\eh@hookend
        \undef\eh@tempi
}
\def\eh@popposthook#1#2\eh@hookseparator#3\eh@hookseparator#4\eh@hookend{%
        \def\eh@tempi{#4}%
        \ifdefequal\eh@sentinel\eh@tempi%
                {\gdef#1{#2}\undef\eh@tempi}%
             {\eh@popposthook#1{#2\eh@hookseparator#3}\eh@hookseparator#4\eh@hookend}%
}
\def\eh@sentinel{\eh@sentinel}
```

`\eh@clearhook`

Internal hook reset.

```
\def\eh@clearhook#1{%
        \global\csdef{eh@#1}{}%
}
```

`\ClearPreHook`
`\ClearPostHook`

Reset the pre/post hook to empty.

```
\newrobustcmd\ClearPreHook[1]{%
        \eh@checkhook{#1}\ClearPreHook
        \eh@clearhook{pre#1}%
}
\newrobustcmd\ClearPostHook[1]{%
        \eh@checkhook{#1}\ClearPostHook
        \eh@clearhook{post#1}%
}
```

`\eh@savehook`
`\eh@restorehook`

Internal macros to `\let` the hook to the supplied control sequence to save. Perform the `\let` in the other direction to restore.

```
\def\eh@savehook#1#2{%
        \letcs#2{eh@#1}%
}
\def\eh@restorehook#1#2{%
```

```
        \global\cslet{eh@#1}#2%
}
```

\SavePreHook
\SavePostHook
\RestorePreHook
\RestorePostHook

User macros to save and restore hooks.

```
\newrobustcmd\SavePreHook[2]{%
        \eh@checkhook{#1}\SavePreHook
        \eh@savehook{pre#1}#2%
}
\newrobustcmd\SavePostHook[2]{%
        \eh@checkhook{#1}\SavePostHook
        \eh@savehook{post#1}#2%
}
\newrobustcmd\RestorePreHook[2]{%
        \eh@checkhook{#1}\RestorePreHook
        \eh@restorehook{pre#1}#2%
}
\newrobustcmd\RestorePostHook[2]{%
        \eh@checkhook{#1}\RestorePostHook
        \eh@restorehook{post#1}#2%
}
```

\eh@saveclearallhooks
\eh@restoreallhooks

Internal macros to save and clear (resp. restore) all hooks at the start (resp. end) of the output routine.

```
\def\eh@saveclearallhooks{%
        \global\eh@savehook{prepar}\eh@or@prepar
        \global\eh@savehook{postpar}\eh@or@postpar
        \global\eh@savehook{premath}\eh@or@premath
        \global\eh@savehook{postmath}\eh@or@postmath
        \global\eh@savehook{predisplay}\eh@or@predisplay
        \global\eh@savehook{postdisplay}\eh@or@postdisplay
        \global\eh@savehook{prehbox}\eh@or@prehbox
        \global\eh@savehook{posthbox}\eh@or@posthbox
        \global\eh@savehook{prevbox}\eh@or@prevbox
        \global\eh@savehook{postvbox}\eh@or@postvbox
        \global\eh@savehook{precr}\eh@or@precr
        \global\eh@savehook{postcr}\eh@or@postcr
        \eh@clearhook{prepar}%
        \eh@clearhook{postpar}%
        \eh@clearhook{premath}%
        \eh@clearhook{postmath}%
        \eh@clearhook{predisplay}%
        \eh@clearhook{postdisplay}%
        \eh@clearhook{prehbox}%
        \eh@clearhook{posthbox}%
        \eh@clearhook{prevbox}%
        \eh@clearhook{postvbox}%
        \eh@clearhook{precr}%
        \eh@clearhook{postcr}%
}
\def\eh@restoreallhooks{%
        \eh@restorehook{prepar}\eh@or@prepar
        \eh@restorehook{postpar}\eh@or@postpar
        \eh@restorehook{premath}\eh@or@premath
        \eh@restorehook{postmath}\eh@or@postmath
        \eh@restorehook{predisplay}\eh@or@predisplay
        \eh@restorehook{postdisplay}\eh@or@postdisplay
        \eh@restorehook{prehbox}\eh@or@prehbox
        \eh@restorehook{posthbox}\eh@or@posthbox
        \eh@restorehook{prevbox}\eh@or@prevbox
        \eh@restorehook{postvbox}\eh@or@postvbox
        \eh@restorehook{precr}\eh@or@precr
        \eh@restorehook{postcr}\eh@or@postcr
}
\endinput
```

Change History

v1.0

General: Initial version 6

v1.1

General: Add excludeor 6

v1.2

General: Let everyX be redefined 6

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.