

Package `paracol`: Yet Another Multi-Column Package to Typeset Columns in *Parallel*

Hiroshi Nakashima (Kyoto University, deceased)
Markus Kurtz (<https://codeberg.org/mgkurtz/paracol>)

version 1.36: 2024-10-22

Contents

I	User's Manual	5
1	Introduction	5
2	Basic Usage	6
3	Column Synchronization	7
4	Environments for Columns	7
4.1	Environment <code>column</code>	7
4.2	Environment <code>nthcolumn</code>	8
4.3	Environments <code>leftcolumn</code> and <code>rightcolumn</code>	8
5	Floats, Footnotes and Counters	9
5.1	Figures and Tables	9
5.2	Footnotes and Marginal Notes	10
5.3	Local and Global Counters	10
6	Closing <code>paracol</code> Environment and Page Flushing	12
7	Reference Manual	15
7.1	Environment <code>paracol</code>	15
7.2	Column-Switching Command and Environments	16
7.3	Commands for Column and Gap Width	19
7.4	Commands for Two-Sided Typesetting and Marginal Note Placement	21
7.5	Commands for Counters	23
7.6	Page-Wise Footnotes	24
7.7	Commands for Coloring Texts and Column-Separating Rules	25
7.8	Commands for Background Painting	27
7.9	Control of Contents Output	29
7.10	Page Flushing Commands	29
8	Numbering and Placement of Page-Wise Footnotes	30
8.1	Multiple <code>\switchcolumn</code> in a Page	30
8.2	Commands <code>\footnote*</code> and Relatives	32
8.3	Page Break	33
9	Two-Sided Typesetting and Parallel-Paging	38
9.1	Example of Paired Parallel-Paging	41
9.2	Example of Non-Paired Parallel-Paging	44
10	Examples of Background Painting	48

10.1	Fundamental Painting	48
10.2	Mirrored Painting and Enlarging/Shrinking/Shifting Regions	52
10.3	Regions with Infinite Extensions	56
11	Known and Unknown Problems	59
II	Implementation	64
1	Overview	64
1.1	Column-Pages	64
1.2	Current Column-Pages and Their Contexts	64
1.3	Pages and Their Contexts	66
1.4	Counters	68
1.5	Page-Wise and Merged Footnotes	68
1.6	Text Coloring	70
1.6.1	Fundamental Mechanism	70
1.6.2	Coloring in Horizontal Mode	71
1.6.3	Changing Default Column Color	72
1.6.4	Coloring in Math Mode	73
1.6.5	Emptiness of a Column-Page	74
1.7	Parallel-Paging, Column-Swapping, Column-Separating Rule Drawing and Background Painting	74
1.8	Page-wise Float Placement	79
2	Interaction with \TeX and \LaTeX	80
2.1	Registers	80
2.1.1	Insertion Registers	80
2.1.2	Integer Registers	81
2.1.3	Dimension Registers	87
2.1.4	Skip Registers	97
2.1.5	Box Registers	99
2.1.6	Token Registers	102
2.1.7	Switches	102
2.2	Macros	105
2.2.1	Procedural Macros	105
2.2.2	Structural Macros	118
3	Register Declaration	128
3.1	$\backslash\text{count}$ Registers	128
3.2	Switches	131
3.3	$\backslash\text{dimen}$ and $\backslash\text{skip}$ Registers	137
3.4	$\backslash\text{box}$ Registers	138
3.5	$\backslash\text{insert}$ Register Set	140
3.6	$\backslash\text{toks}$ Register	140
4	Logging Tools	140
5	$\backslash\text{output}$ Routine	142
6	Completing Column-Page	144
7	Starting New Page	152
8	Shipping Page Out	154
9	Starting New Column-Page	164
10	Background Painting	165
11	Special Output Routines	173
11.1	Dispatcher	173
11.2	Building Starting Page	174
11.3	Column-Switching	180

11.4 Color Management	187
11.5 Footnote Handling	192
11.6 Marginal Notes	194
11.7 Synchronization	201
11.8 Page Flushing	214
11.9 Last Page	222
12 Starting Environment	226
13 Column Width Setting	234
14 Counter Operations	242
15 Column-Switching Commands and Environments	246
16 Disabling <code>\addcontentsline</code>	250
17 Page Flushing Commands	252
18 Commands for Footnotes	253
19 Commands for Marginal Notes	257
20 Two-Sided Typesetting	258
21 Commands for Text Coloring	260
22 Commands for Column-Separating Rule Color and Background Painting	264
23 Closing Environment	268
Acknowledgments	270
Index	271
Revision History	289

PART I

User's Manual

Abstract

This package provides a L^AT_EX environment named `paracol` in which you may *switch* and *synchronize* columns by a command `\switchcolumn` and by internal environments `column`, `nthcolumn`, `leftcolumn` and `rightcolumn`.

1 Introduction

This document describes the usage of yet another multi-column package named `paracol`. The unique feature of the package is that columns are typeset *in parallel*.

Suppose you are writing a bilingual document whose left column is written in a language, say English, and right column has the translation of the left column in another language, e.g., Japanese. With the `paracol` package you may write an English part of arbitrarily length and then *switch* to its Japanese counterpart to place both parts side by side. Of course you may return to the English writing similarly.

The *column-switching* is always allowed when you complete an outermost level paragraph. You may be unaware whether a column is broken into multiple pages before switching because the package automatically goes back and forward to the correct page and vertical position when you switch the column. Moreover, you may *synchronize* columns so that the tops of the first paragraphs after switching in all columns are vertically aligned. At a synchronization point, you may give a single-column text, for example a common section header, optionally. You may also switch single-column and multi-column in a page arbitrary.

This manual itself is an example of two-column

```
\begin{paracol}{2}[\section{Introduction}]
\hbadness5000
This document describes the usage of yet
another multi-column package named
\textsf{paracol}. The unique feature of
the package is that columns are typeset
{\em in parallel.}
```

Suppose you are writing a bilingual document whose left column is written in a language, say English, and right column has the translation of the left column in another language, e.g. Japanese. With the `\textsf{paracol}` package you may write an English part of arbitrary length and then `{\em switch}` to its Japanese counterpart to place both parts side by side. Of course you may return to the English writing similarly.

The column switching is always allowed when you complete an outermost level paragraph. You may be unaware whether a column is broken into multiple pages before switching because the package automatically goes back and forward to the correct page and vertical position when you switch the

documents typeset by `paracol`. Since the author is not familiar with languages other than English and Japanese and the latter should be hardly understood by most of readers, the right column is the translation of the left English column into a computational language. That is, the right column is the `LATEX` source code of the left column¹.

Moreover, you may `{\em synchronize}` columns so that the tops of the first paragraphs after switching in all columns are vertically aligned. At a synchronization point, you may give a single-column text, for example a common section header, optionally. You may also switch single-column and multi-column in a page arbitrary.

This manual itself is an example of two-column documents typeset by `\textsf{paracol}`. Since the author is not familiar with languages other than English and Japanese and the latter should be hardly understood by most of readers, the right column is the translation of the left English column into a computational language. That is, the right column is the `\LaTeX{}` source code of the left column%
`\footnote{Not really but its essence is shown.}`.

```
\switchcolumn
```

```
\begin{verbatim}
Here is the source of above.
\end{verbatim}1
```

2 Basic Usage

Loading the package is very simple. What you have to do is `\usepackage{paracol}` in the preamble. Or write `\usepackage{paracol}[=2018-12-31]` to get the last version authored by Hiroshi Nakashima, if this is your wish.

The fundamental means of parallel-column typesetting are the environment `paracol` and the command `\switchcolumn`. The `paracol` environment needs an argument to specify the number of columns. Thus the following is the basic construct for two-parallel-column documents.

```
\begin{paracol}{2}
left column text
\switchcolumn
right column text
```

¹Not really but its essence is shown.

```
\switchcolumn*[\section{Basic Usage}]
Loading the package is very simple. What you have to do is |\usepackage{paracol}| in the preamble. ...2
```

```
\switchcolumn
```

source

```
\switchcolumn*
```

The fundamental means of parallel-column typesetting are the environment `|paracol|` and the command `|\switchcolumn|`. ...

```
\switchcolumn
```

source

¹This `verbatim` construct is simply referred as to “*source*” hereafter.

²Hereafter, a part of the source code may be omitted like this.

```

\switchcolumn
left column text
\switchcolumn
right column text
\switchcolumn
:
\end{paracol}

```

The `\switchcolumn` command may have an optional argument to specify the column number (zero origin) to start. That is, `\switchcolumn[0]` means to switch to the leftmost column, `\switchcolumn[1]` is to start the second column and so on. Thus the `\switchcolumn` without the optional argument may be considered as `\switchcolumn[$i+1 \bmod n$]` where i is the ordinal of the column you are leaving from and n is the number of columns given to `paracol` environment.

3 Column Synchronization

The `\switchcolumn` command may also be followed by a ‘*’ to *synchronize* columns. After you switch from a column to another by `\switchcolumn*` (or `\switchcolumn[i]*`), all the columns are vertically aligned at the bottom of the *deepest* one preceding the command. For example, the previous section has three `\switchcolumn*` commands at which left and right columns are vertically aligned.

The *starred* version of `\switchcolumn` may have an optional argument to specify a single-column *spanning text* whose bottom is the vertical alignment point of columns. For example, `\section` commands in this manual are given as optional arguments of `\switchcolumn*` like;

```
\switchcolumn*[\section{Basic Usage}]
```

The `paracol` environment may also start with a spanning text by specifying it as the optional argument of `\begin{paracol}`. For example, at the beginning of this document, the author put;

```
\begin{paracol}{2}[\section{Introduction}]
```

4 Environments for Columns

4.1 Environment column

The `\switchcolumn` is simple but you may prefer to pack the contents of a column in an environment. The `column` environment is available for this

```
\switchcolumn[1]*
```

```
source
```

```
\switchcolumn[0]
```

The `|\switchcolumn|` command may have an optional argument to specify the column number (zero origin) to start. ...

```
\switchcolumn[0]*[%
```

```
\section{Column Synchronization}
```

```
\label{sec:sync}]
```

The `|\switchcolumn|` command may also be followed by a ‘|*|’ to `{\em synchronize}` columns. ...

The `{\em starred}` version of

`|\switchcolumn|` may have an optional argument to specify a multi-column text whose bottom is the vertical alignment points of the columns. ...

```
\switchcolumn
```

```
source
```

4.1 Environment column

```
\begin{column*}[%
```

```
\section{Environments for Columns}
```

```
\label{sec:env}]
```

well-structuralization of L^AT_EX sources for parallel-columned documents. A construct;

```
\begin{column}  
  text for a column  
\end{column}
```

is (almost) equivalent to;

```
\switchcolumn  
  text for a column
```

The `column*` environment is also available for the column synchronization and may have an optional argument for spanning text.

4.2 Environment `nthcolumn`

The `\switchcolumn` can start an arbitrarily specified column with the column number given through its optional argument, but the `column` environment cannot do it. If you want to start *i*-th column, you have to do `\begin{nthcolumn}{i}` (or `nthcolumn*` with an optional argument to synchronize).

4.3 Environments `leftcolumn` and `rightcolumn`

The environments `leftcolumn` and `rightcolumn` (and their starred versions with an optional argument) are available as more convenient means than saying `\begin{nthcolumn}{0}` to switch to the left(most) column and `\begin{nthcolumn}{1}` to the right (but may not be rightmost) one.

```
\subsection{Environment \texttt{column}}  
The |\switchcolumn| is simple but you may prefer to pack the contents of a column in an environment. ...  
\end{column*}  
\begin{column}  
  source  
\end{column}
```

4.2 Environment `nthcolumn`

```
\begin{nthcolumn*}{1}  
  source  
\end{nthcolumn*}  
  
\begin{nthcolumn}{0}  
\subsection{Environment \texttt{nthcolumn}}  
The |\switchcolumn| can start an arbitrarily specified column with the column number given through its optional argument, but the |column| environment cannot do it. ...  
\end{nthcolumn}
```

4.3 Environment `leftcolumn` and `rightcolumn`

```
\begin{leftcolumn*}  
\subsection{%  
  Environments \texttt{leftcolumn} and \\  
  \texttt{rightcolumn}}  
The environments |leftcolumn| and |rightcolumn| (and their starred versions with an optional argument) are available as more convenient means than saying |\begin{nthcolumn}{0}| to switch to the left(most) column and ...  
\begin{figure*}...\end{figure*}  
\begin{figure}[t]...\end{figure}  
\end{leftcolumn*}  
\begin{rightcolumn}  
  source and a figure environment  
\end{rightcolumn}
```


double-column figure #1

Figure 1: A Double-Column Figure

single-column figure #1

Figure 2: A Single-Column Figure

single-column figure #2

Figure 3: Another Single-Column Figure

5 Floats, Footnotes and Counters

5.1 Figures and Tables

As shown in this page, double-column figures/tables (or those spanned multiple columns if you have three or more) may be placed by `figure*` and `table*` environments as usual². A single-column figure/table will be placed in the column in which you put `figure` and `table`. For example, the body of a `figure` environment in a `leftcolumn` environment is *always* placed in a left column. That is, even if the column of the *current* page does not have enough room to place the figure, it will not be thrown to the right column but will be placed in the left column of the next page³.

Another caution about float placement is that you have to be careful when you try to put a top-float explicitly with `t`-option or implicitly without placement option (i.e., `tbp` in most classes) and to synchronize columns. The rule is as follows; after you synchronize columns in a page, the page cannot have top-floats any more. When you synchronize columns, `paracol` fixes a virtual horizontal line in the page as the synchronization barrier. Thus no top-floats can-

²See Section 11 for the appearance order issue of double-column floats.

³Or some farther page if L^AT_EX cannot solve the placement problem wisely.

Table 1: A Single-Column Table

An	example	of
single	column	table

5.1 Figures and Tables

```
\begin{leftcolumn*}[\section{%
  Floats, Footnotes and Counters}]
\begin{table}[b]
\caption{A Single-Column Table}
\centerline{\begin{tabular}[t]{|l|c|r|}
\hline
An&example&of\\\hline
single&column&table\\\hline
\end{tabular}}
\end{table}
```

`\subsection{Figures and Tables}`
As shown in this page, double-column figures/tables (or those spanned multiple columns if you have three or more columns) may be placed by `figure*` and `table*` environments as usual³. ...

5.2 Footnotes and Marginal Notes

Footnotes are also put at the bottom of the column in which `|\footnote|` commands and their references reside (like `this\footnote{...}`), as shown in page~2 and this page. Marginal notes behave similarly

³Another example of footnote.

Table 2: Another Single-Column Table

Another	example
of	single
column	table

not be added above the line⁴. Therefore, the author put two `figure` environments for the figures shown in this page into the `leftcolumn*` and `rightcolumn` environment for the previous section.

5.2 Footnotes and Marginal Notes

Footnotes are also put at the bottom of the column in which `\footnote` commands and their references reside (like this⁵), as shown in page 6 and this page.

Marginal notes behave similarly like what you are

An example seeing in the left margin of this sentence and the of marginal right marginal note in this page⁶.
note.

5.3 Local and Global Counters

You probably found that the numbering of figures and tables is *global* while that of footnotes are *local*. That is, the figure in the right column of the previous page has number 3 following its left-column counterpart Figure 2. The tables in the page are also numbered as 1 and 2 crossing the column boundary. However, the footnotes in each column have their own numbering sequence. Moreover, the footnote numbers in left columns are typeset in roman font while those in right columns have italic shapes. Similarly, subsection numbering is local and the headings in right columns have typewriter-face numbers.

This happens because the author declared the counters `figure` and `table` are *global* in the preamble of this document by saying;

```
\globalcounter{figure}
\globalcounter{table}
```

and do nothing about `footnote` and `subsection` counters. By default, all the counters except for `page` are local to columns. The value of a local counter of a column is saved somewhere when you leave the column, and it is restored when you revisit the column. The initial values of the local counters are the values they have at `\begin{paracol}`. After you close the `paracol` environment, the values of the leftmost

⁴Even if you have enough space above, sorry.

⁵Unless you specify to make footnotes *page-wise* as explained in Section 7.6 and 8.

⁶If you have three or more columns, marginal notes of the second or succeeding columns are placed in the right margin in default setting. The `paracol` package solves the placement problem of marginal notes from two or more columns sharing a side margin by moving some of them down if they conflict over the space with each other.

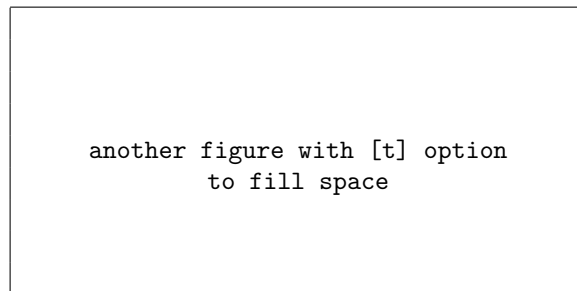


Figure 4: Another Figure with [t] Option

like what you are seeing in the left margin of this sentence\marginpar{\raggedright An example of marginal note.} and the right marginal note in this page\footnote{...}. ...

Another example of marginal note.

5.3 Local and Global Counters

You probably found that the numbering of figures and tables is `\emph{global}` while that of footnotes are `\emph{local}`. ...

```
\end{leftcolumn*}
\begin{rightcolumn}
source.
\end{rightcolumn}
```

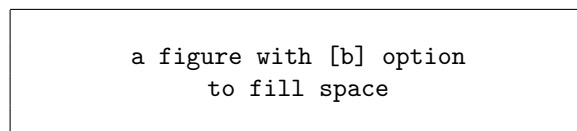


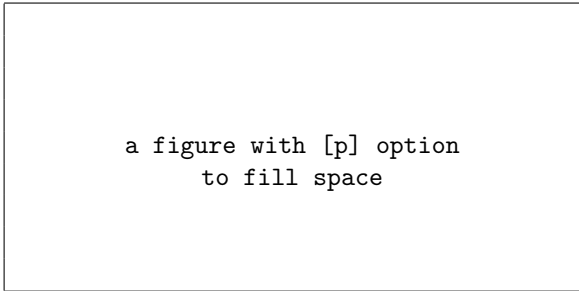
Figure 5: A Figure with [b] Option

column are used for the rest of your document until you start new `paracol` environment. On a restart, local counters in a column have the values they had at the last `\end{paracol}`, except for those which have been modified outside the environment because the modifications are *broadcasted* to local counters in all columns. You will see the effect of this inter-environment counter value conservation in the footnote numbers in the right column in page 9 and 13.

This broadcasting of a local counter value can be done explicitly in `paracol` environments by a command `\synccounter{ctr}`. This command makes `ctr` in all columns have the value of that in the column in which the command appears. In addition, another command `\syncallcounters` performs this broadcasting for all local counters.

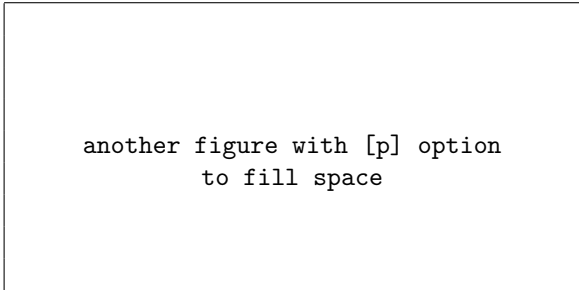
If you make a counter global by the command `\globalcounter`, the save/restore operations are not performed to the counter and thus it is globally incremented by `\[ref]stepcounter` or commands such as `\caption` and `\section`. Note that the value of a global counter depends on the place where it is incremented (or set) in the *source code* rather than where it appears in the output. Thus if the author put a `table` environment here to increment `table` counter, the right-column table at the bottom of page 9 would be Table 3 because its `table` environment does not appear yet in the source code. Note that, however, though the counter `page` is global as expected, its numbering is consistent among all columns as far as you refer to the value by `\pageref{label}` and/or see the values in table of contents, etc.

Another counter which the author made global in this document is `section`. As explained in Section 3, an optional spanning text of column-switching is considered as in the leftmost column. Since `\section` commands in this document are always given in spanning texts, so far, it seems unnecessary to make `section` global because it is incremented correctly in the leftmost column. However, the stepping `section` has a side effect to reset its descendent counter `subsection` and referred to from `\thesubsection` command. Thus if `section` were local, the right-column subsections in Section 4 would be numbered as “0.1”, “0.2” and “0.3” because the local value of `section` would be zero. Moreover, the right-column subsections of this section would be “0.4”, “0.5” and “0.6” because stepping `section` local to the left column would not reset `subsection` local to the right column.



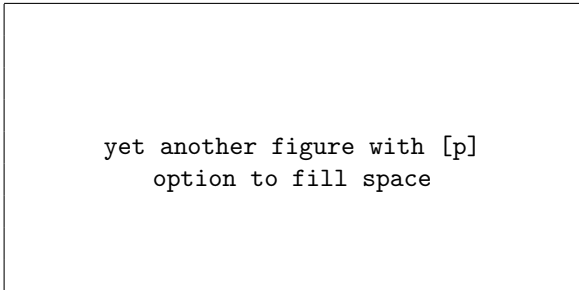
a figure with [p] option
to fill space

Figure 6: A Figure with [p] Option



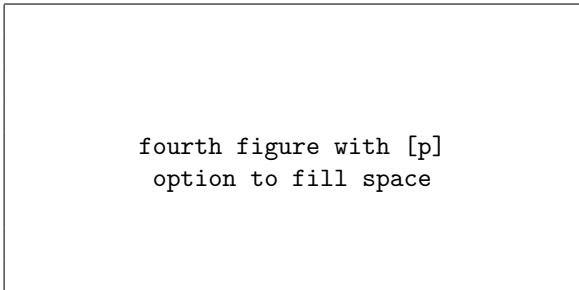
another figure with [p] option
to fill space

Figure 7: Another Figure with [p] Option



yet another figure with [p]
option to fill space

Figure 8: Yet Another Figure with [p] Option



fourth figure with [p]
option to fill space

Figure 9: Forth Figure with [p] Option

You may give a local appearance to a counter *ctr* for the *i*-th column (zero origin) by a command;

```
\definethecounter{ctr}{i}{def}
```

where *def* is to be the body of the local definition of `\thectr`. For example, the preamble of this document has the following to give non-default definitions to `\thefootnote` and `\thesubsection` for right columns.

```
\definethecounter{footnote}{1}{%
  \textit{\arabic{footnote}}}
\definethecounter{subsection}{1}{%
  \texttt{%
    \arabic{section}.\arabic{subsection}}}
```

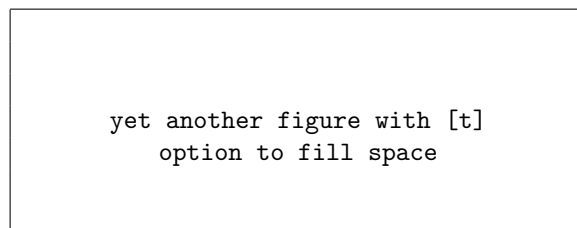


Figure 10: Yet Another Figure with [t] Option

6 Closing paracol Environment and Page Flushing

The final example shown here is this single-column text which the author put after the `paracol` environment above is closed. As you are seeing, a `paracol` environment can be finished at any vertical position in a page and can be followed by ordinary single column texts.

The environment may also be restarted anywhere you like as shown here.

The last issue is to flush a page. The ordinary `\newpage` command works as you expect. If you say `\newpage` in the left column in a page, the contents following it will appear in the left column in the next page. Note that this does not affect the layout of the right column.

To flush all columns in a page, a command `\flushpage` is available. This command in *i*-th column is almost equivalent to;

```
\switchcolumn[i]*[\newpage]
```

but more robust⁷. The ordinary page breaking command `\clearpage` may also be used to flush all columns and to start a fresh page, but it has a side effect to put all figures and tables which are not yet output.

Now the author will do `\flushpage` shortly to start a real binlingual example from the next page, after showing another example of closing `paracol` environments in this sentence and of restarting in the next one, in which *unbalanced column width* is demonstrated using `\columnratio` command shown in Section 7.3.

O.K., we have restarted `paracol` environment and we will see the effect of `\flushpage` now!!

```
\begin{paracol}{2}
\begin{leftcolumn}
The environment may also be restarted
anywhere you like as shown here. ...
\end{leftcolumn}
\begin{rightcolumn}
source
\end{rightcolumn}
\end{paracol}
Now the aurthor will do ...
```

```
\columnratio{0.6}
\begin{paracol}{2}
\begin{leftcolumn}
O.K., ...
\end{leftcolumn}
\begin{rightcolumn} source
\end{rightcolumn}
```

⁷For example `\switchcolumn*` may flush a page for the synchronization and thus `\newpage` may leave an empty page.

An Die Freude/To Joy Friedrich Schiller

The following is the libretto of the fourth movement of Beethoven's Ninth Symphony, his adaptation of Schiller's ode "An Die Freude" (or "To Joy" in English). Beethoven's additions and revisions are indicated in italics.

O Freunde, nicht diese Töne!
Sondern laßt uns angenehmere anstimmen und freudenvollere
⁸.

Oh friends, no more of these sad tones!
Let us rather raise our voices together
*In more pleasant and joyful tones*⁴.

Freude!
Freude, schöner Götterfunken Tochter aus Elysium,
Wir betreten feuertrunken, Himmlische, dein Heiligtum!
Deine Zauber binden wieder, *Was die Mode streng geteilt;*
*Alle Menschen werden Brüder*⁹, Wo dein sanfter Flügel weilt

Joy!
Joy, thou shining spark of God,
Daughter of Elysium,
With fiery rapture, goddess,
We approach thy shrine.
Your magic reunites
That which stern custom has parted;
*All humans will become brothers*⁵
Under your protective wing.

Wem der große Wurf gelungen, eines Freundes Freund zu sein;
Wer ein holdes Weib errungen, mische seinen Jubel ein!
Ja, wer auch nur eine Seele sein nennt auf dem Erdenrund!
Und wer's nie gekonnt, der stehle weinend sich aus diesem
Bund!

Let the man who has had the fortune
To be a helper to his friend,
And the man who has won a noble woman,
Join in our chorus of jubilation!
Yes, even if he holds but one soul
As his own in all the world!
But let the man who knows nothing of this
Steal away alone and in sorrow.

Freude trinken alle Wesen an den Brüsten der Natur;
Alle Guten, alle Bösen folgen ihrer Rosenspur.
Küsse gab sie uns und Reben, einen Freund, geprüft im Tod;
Wollust ward dem Wurm gegeben, und der Cherub steht vor
Gott.

All the world's creatures drink
From the breasts of nature;
Both the good and the evil
Follow her trail of roses.
She gave us kisses and wine
And a friend loyal unto death;
She gave the joy of life to the lowliest,
And to the angels who dwell with God.

Froh, wie seine Sonnen fliegen durch des Himmels prächt'gen
Plan,
Laufet, Brüder, eure Bahn, freudig, wie ein Held zum Siegen.

Joyous, as his suns speed
Through the glorious order of Heaven,
Hasten, brothers, on your way,
Joyful as a hero to victory.

⁸If I had been a good student in my German class, I could find the German translation of the right column footnote ⁴ is "Dieser Teil wurde von Beethoven hinzugefügt" by myself without the kind help from a user.

⁹Original: Was der Mode Schwert geteilt;
Bettler werden Fürstenbrüder,

⁴This part was added by Beethoven.

⁵Original: What custom's sword has parted;
Beggars become princes' brothers

Seid umschlungen, Millionen! Diesen Kuß der ganzen Welt!
Brüder, über'm Sternenzelt muß ein lieber Vater wohnen.

Ihr stürzt nieder, Millionen? Ahnest du den Schöpfer, Welt?
Such'ihn überm Sternenzelt! Über Sternen muß er wohnen.

Be embraced, all ye millions!
With a kiss for all the world!
Brothers, beyond the stars
Surely dwells a loving Father.

Do you kneel before him, oh millions?
Do you sense the Creator's presence?
Seek him beyond the stars!
He must dwell beyond the stars.

7 Reference Manual

7.1 Environment `paracol`

`\begin{paracol}{num}[text] body \end{paracol}`

The environment `paracol` contains *body* typeset in *num* columns in parallel. The optional *text* is put spanning all columns prior to the multi-columned *body*.

- The environment may start from *any* vertical position in a page, i.e., not necessary at the top of a page. The single-column *pre-environment stuff* of the *starting page* in which `\begin{paracol}` lies are naturally connected to the beginning part of *body* in each column, unless the page has footnotes¹⁰ or bottom floats. If these kinds of bottom stuff exist, they are put above the multi-columned *body*, or the spanning *text* if provided, with a vertical skip of `\textfloatsep` separating them if bottom floats exist, or of `\belowfootnoteskip` described in Section 7.6 if only footnotes exist. The *deferred* floats which have not yet appeared in the starting page and thus will appear in the next or succeeding pages are considered as page-wise floats given in the environment.
- The environment can be enclosed in a *list-like environment* such as `enumerate`, `itemize` and `description`. If so, `\items` in each column are typeset using the parameters of the surrounding environment such as `\leftmargin` and `\rightmargin`. For example, the following short `paracol` environment is included in an `itemize` for this and other `\items` in this page.
- This is the first `\item` in the left column.
- This is the second `\item` in the left column followed by a `\switchcolumn`¹¹.
- This is the first `\item` in the right column.
- This is the second `\item` in the right column.
- This is the third and last `\item` in the right column.

You are now seeing the switching to/from multi-columned and `itemized` texts are naturally connected with the last and this single-columned sentences. You may feel the space between two columns above is too large but it simply results from the large total `\leftmargins` of the outer `description` and this `itemize`, which make the right column shifted right. A simple remedy for this large space is to make `\columnsep` narrower, for example 0pt as shown below.

- This `\item` is wider than the last `\item` above because `\columnsep` is 0pt.
- Therefore, this `\item` is shifted left a little bit to make inter-column space narrower.
- All local counters in all columns are initialized to have the values at `\begin{paracol}` on its first occurrence. On the second and succeeding occurrences of `\begin{paracol}`, the local counters in each column have the value at the last `\end{paracol}`, unless they are modified after the `\end{paracol}`. If a counter is modified (or declared by `\newcounter`) after the `\end{paracol}`, the local versions of the counter in all columns commonly have the value at `\begin{paracol}`.
- The environment may end at *any* vertical position in a page, i.e., the *post-environment stuff* being the single-column texts and others following `\end{paracol}` in the *last page* of the environment may not start from the top of a page. If any columns don't have deferred column-wise floats and the most advanced *leading column* at `\end{paracol}` has neither of footnotes¹² nor bottom floats, its bottom is naturally connected to the post-environment stuff. If the leading column has these kinds of bottom stuff, they are put above the post-environment stuff, with a vertical skip of `\textfloatsep`

¹⁰With merged footnote layout shown in Section 7.6, the footnotes in the single-column contents are merged with those in `paracol` environment and are put at the bottom of the starting page together as shown in this page.

¹¹This footnote is to show the footnotes in this page are merged.

¹²With merged footnote layout shown in Section 7.6, the footnotes in the closing `paracol` environment are merged with those in post-environment stuff and are put at the bottom of the page together as shown in this page.

separating them if bottom floats exist. All deferred column-wise floats given in the environment are flushed before the post-environment stuff appears, possibly creating *float columns* only with floats. On the other hand, deferred page-wise floats given in the environment are considered as deferred (single-) column-wise floats given just after `\end{paracol}`.

- The values of all local counters in the leftmost column are used as the initial values of them in the post-environment stuff.
- The `paracol` environment cannot be nested, or you will have an error message of illegal nesting.
- The commands `\switchcolumn`, `\synccounter`, `\syncallcounters` and `\flushpage`, and environments `column(*)`, `nthcolumn(*)`, `leftcolumn(*)` and `rightcolumn(*)` are *local* to `paracol` environment and thus undefined outside the environment¹³. The command `\clearpage` is of course usable outside and inside the environment but its function inside is a little bit different from outside.

```
\begin{paracol}[numleft]{num}[text] body \end{paracol}
\begin{paracol}[numleft]*{num}[text] body \end{paracol}
```

If a `\begin{paracol}` has the optional *numleft* argument to specify the number of leading columns n_l together with the total n given by *num*, columns in the environment are laid out across two adjacent pages. In this *parallel-page* typesetting, the first n_l columns are placed in the *left* page while remaining $n_r = n - n_l$ columns go to the next *right* page. The pair of left and right pages is considered as comprising a virtual *paired* page and thus shares a common page number, unless *non-paired* typesetting is specified by the optional ‘*’ following the optional *numleft* argument. In the non-paired parallel-paging, when the leading n_l columns are put in a page p , the trailing n_r columns are in the page $p + 1$.

- All *page-wise stuff*, i.e., pre-environment and post-environment stuff, page-wise floats, spanning text and (merged or non-merged) page-wise footnotes, are placed only in left parallel-pages leaving corresponding regions in right parallel-pages blank¹⁴.
- A non-paired left parallel-page is not necessary to be even-numbered, though the printing tradition requires so if you naturally want to have a parallel-page pair in a double spread. The page number given to the first left parallel-page is simply the number of the page p_1 in which `\begin{paracol}` reside, and that for the k -th left parallel-page is $p_1 + 2(k - 1)$ ¹⁵. Therefore, to make it sure p_1 is even, you might need to have an ordinary page of blank, a title, etc., or to let `page` counter have an even number by `\setcounter`, etc., before starting a `paracol` environment.
- Section 9 shows examples of parallel-paging together with related issues on two-sided typesetting.

7.2 Column-Switching Command and Environments

```
\switchcolumn[col]
\switchcolumn[col]*[text]
```

The command switches columns from i to j where i and j is the zero-origin ordinals of the columns from/to which we are leaving/visiting respectively. Without the optional *col*, $j = i + 1 \bmod n$ where n is the number of columns given to `\begin{paracol}`, while $j = col$ with the optional argument. If the command (or *col* if specified) is followed by a *, the column-switching takes place after synchronization and, if specified, the optional spanning *text* is put.

- Using `\switchcolumn` in a list-like environment *included* in a `paracol` environment causes an ugly result without any error/warning messages. This caution is effectual for all column-switching environments too.

¹³Unless you dare to define them.

¹⁴Someday the author could devise an advanced mechanism to exploit the space in right parallel-pages.

¹⁵Unless you make some change to `page` counter.

- If $col \notin [0, n)$, an error is reported and, if you dare to continue, you will switch to the leftmost column 0.
- The synchronization point is set just below the last line of the leading column in a page p , partly taking deferred floats into account. That is, all deferred floats are put in the pages up to $p - 1$ and at the top of p if possible. Then, if a non-leading column has footnotes and/or bottom floats and they cannot be pushed down below the synchronization point, the point is moved to the next page top¹⁶.
- In a page having one or more synchronization points, stretch and shrink factors of all vertical spaces, such as those surrounding sectioning commands, are ignored. Therefore, even if you specify `\flushbottom`, the page is typeset as if `\raggedbottom` were specified.
- After a synchronization point is set, no top floats will be inserted in the page having the point, thus they will be deferred to the next page or further one.

```
\begin{column}  body  \end{column}
\begin{column*}[text]  body  \end{column*}
```

The environment `column` contains *body* for the column next to what we are in just before `\begin{column}`. The starred version `column*` does the same after synchronization and, if specified, the optional spanning *text* is put.

- The environments are almost equivalent to;

```
{\switchcolumn  body  \par}
{\switchcolumn*[text]  body  \par}
```

except for their first occurrences which don't switch to the column 1 (i.e., right column if two-columned) but stay in the leftmost column 0. More precisely, `\begin{column(*)}` does not make column-switching if it is not preceded by `\switchcolumn` nor other column-switching environments.

- The *body* of the environments cannot have `\switchcolumn` nor column-switching environments including `column(*)` themselves, or you will have an error message of illegal use of command/environment.
- Column-switching does not take place at `\end{column(*)}`. Therefore, texts following the environments are put in the column in which *body* resides until a column-switching command/environment is given.

```
\begin{nthcolumn}{col}  body  \end{nthcolumn}
\begin{nthcolumn*}{col}[text]  body  \end{nthcolumn*}
```

The environment `nthcolumn` contains *body* for the column *col*. The starred version `nthcolumn*` does the same after synchronization and, if specified, the optional spanning *text* is put.

- The environments are equivalent to;

```
{\switchcolumn[col]  body  \par}
{\switchcolumn[col]*[text]  body  \par}
```

- The *body* of the environments cannot have `\switchcolumn` nor column-switching environments including `nthcolumn(*)` themselves, or you will have an error message of illegal use of command/environment.
- Column-switching does not take place at `\end{nthcolumn(*)}`. Therefore, texts following the environments are put in the column in which *body* resides until a column-switching command/environment is given.

¹⁶Or below top floats deferred to the page.

```

\begin{leftcolumn} body \end{leftcolumn}
\begin{leftcolumn*}[text] body \end{leftcolumn*}
\begin{rightcolumn} body \end{rightcolumn}
\begin{rightcolumn*}[text] body \end{rightcolumn*}

```

The environment `leftcolumn` contains *body* for the leftmost column 0, while `rightcolumn` for the column 1 being the right column in two-column typesetting. The starred versions `leftcolumn*` and `rightcolumn*` do the same after synchronization and, if specified, the optional spanning *text* is put.

- The environments `leftcolumn(*)` are equivalent to;

```

\begin{nthcolumn}{0} body \end{nthcolumn}
\begin{nthcolumn*}{0}[text] body \end{nthcolumn*}

```

while `rightcolumn(*)` are equivalent to;

```

\begin{nthcolumn}{1} body \end{nthcolumn}
\begin{nthcolumn*}{1}[text] body \end{nthcolumn*}

```

`\thecolumn`

The command gives you the zero-origin ordinal of the column in which this command appears. Therefore, the following code snip;

```

\begin{paracol}{3}
Column-\thecolumn.\switchcolumn Column-\thecolumn.\switchcolumn Column-\thecolumn.
\end{paracol}

```

gives us the followings.

Column-0.

Column-1.

Column-2.

- The command is *neither* a L^AT_EX's counter nor `\count` register of native T_EX, and thus the value it keeps cannot be modified. However, it can be used wherever an integer number is required or appropriate. Therefore for example, `\setcounter{mycounter}{\thecolumn}` works well to give the column ordinal to the counter `mycounter`.

`\definecolumnpreamble{col}{pream}`

The command is to define the column preamble *pream* for the column *col*, which is inserted at every column-switching to the column. More specifically, the command let `\switchcolumn` to *col* act as if you specify;

```

\switchcolumn <pream for col>

```

and column-switching environments such as `nthcolumn` act as if you specify;

```

\begin{nthcolumn}{col} <pream for col>

```

- The optional spanning text of `\switchcolumn`, column-switching environments and `\begin{paracol}` is considered to be in a virtual column -1 , and thus if you need a preamble for spanning texts do `\definecolumnpreamble{-1}{pream}`.
- The command may appear in a `paracol` environment and, if so, *pream* is effective from the succeeding column-switching to *col*.
- The definition of *pream* is made globally.

`\ensurevspace{len}`

The command tells the first synchronizing column-switching command (i.e., `\switchcolumn[col]*`) or environment (i.e., `column*`, etc.) following this command that the page must be broken before synchronization unless the synchronization point has the space of `len` or more below it in the page. If a synchronization does not have the command after the previous synchronization, it is assumed that `\ensurevspace{\baselineskip}` is given.

- This command is to be used when a synchronization point would be placed near the bottom of a page p and the space below it is not sufficient for a column c to put anything in the page, while another column c' can have a few lines in the page. If this happens, the first line after the synchronization should start at the top of the page $p + 1$ in the column c , while that of c' is still in the page p , giving you an impression that the synchronization fails to align the top of all columns below it. The fact is, however, the synchronization point is properly established near at the bottom of the page but the first line of c needs some large space due to, for example, the followings.
 - The line has unusually tall stuff including larger font letters.
 - The line has a footnote reference which is hardly apart from the footnote, and thus the line and the footnote go to the next page together.
 - The parameter `\clubpenalty` is too large (e.g., 10000) to break the first and second lines into separate pages.
 - The first line follows a vertical space.
- This manual itself has some instances of `\ensurevspace` command in the page 13 and 14 in which each German stanza is enclosed in `verse` and then `leftcolumn*` environments and has `\ensurevspace{2\baselineskip}` before the `\begining` of the outer `leftcolumn*` because the first line of the stanza is preceded by a vertical space inserted by `\begin{verse}`. In fact without `\ensurevspace`, the first two lines of the sixth English stanza would be in the page 13, while corresponding German stanza go to the next page 14 as a whole, due to the difference of the height of footnotes in each column, i.e., German ones are taller than English ones to narrow the space for the German column.
- As the author does in the “An die Freude/To Joy” example, it is a good tactics to have an `\ensurevspace` with some vertical space larger than the default `\baselineskip` if it is sure that a column has a feature shown above regardless of the position of the synchronization point in question, because the point goes up or down with revisions of your document and using an `\ensurevspace` for a synchronization far above the page bottom is perfectly harmless. Similarly, if you find a problem in a synchronization and add an `\ensurevspace` to solve it, keeping the command attached is recommended even when the synchronization point moves up or down to make the command unnecessary.

7.3 Commands for Column and Gap Width

`\columnratio{r_0, r_1, \dots, r_k}[r'_0, r'_1, \dots, r'_k]`

The command defines the width of each column by the fraction r_i to specify the portion which i -th ($i = 0$ for the leftmost) column occupies. More specifically, the width w_i of the i -th column is defined as follows, where W is `\textwidth`, S is `\columnsep`, and n is the number of columns given to `\begin{paracol}`.

$$W' = W - (n - 1)S$$
$$w_i = \begin{cases} r_i W' & i \leq k \\ \frac{(1 - \sum_{j=0}^k r_j) W'}{n - (k + 1)} & i > k \end{cases}$$

For a `paracol` environment with parallel-paging, n is replaced with n_l for the columns in left parallel-pages, while n and w_i are replaced with n_r and w_{n_r+i} for those in right parallel-pages. Moreover, if the optional argument having $r'_0, r'_1, \dots, r'_{k'}$ is provided, w_{n_r+i} for a column in right parallel-pages is determined by r'_i and k' instead of r_i and k .

- The equations above imply that $k < n - 1$, $r_i > 0$ and $\sum_{j=0}^k r_j < 1$. If $k \geq n - 1$, k is assumed to be $n - 2$ and all r_i such that $i \geq n - 1$ are ignored. If r_i or its sum does not satisfy the conditions, you will have an ugly result with “Overfull” messages.
- The argument r_0, r_1, \dots, r_k can be empty to mean $k = -1$ to let all column widths be W'/n as default.
- The setting of column width by the command takes effect in the `paracol` environments following the command¹⁷. Therefore, though placing the command in the preamble is the most natural way¹⁸, you may place this command between two `paracol` environments to change the column layout for the second one even when they appear in a page as shown in Section 6.
- In the i -th column, `\columnwidth` has w_i and, for outermost paragraphs in the column, `\hsize` has w_i as well. As for `\linewidth`, it has $w_i - (\text{\textwidth} - l)$ where l is what `\linewidth` had at `\begin{paracol}`, i.e., the `\linewidth` for the list-like environment surrounding `paracol` if any, or `\textwidth` otherwise.
- You can specify width of each column and that of each *gap* between two columns more detailedly by `\setcolumnwidth` shown below. If your document has both of `\columnratio` and `\setcolumnwidth` prior to a `paracol` environment, the command given later is effective for the environment.

`\setcolumnwidth`{ s_0, s_1, \dots, s_k }[$s'_0, s'_1, \dots, s'_{k'}$]

The command defines the width of each column and that of each *gap* between two columns by the column/gap specification s_i for the i -th column and the gap between it and the $(i+1)$ -th column. More specifically, s_i has the form of \hat{w}_i or \hat{w}_i / \hat{g}_i where each of \hat{w}_i and \hat{g}_i is a proper glue including a proper dimension, or an empty string to mean $\hat{w}_i = \text{\fill}$ and $\hat{g}_i = \text{\columnsep}$, to determine the width of i -th column w_i and that of i -th gap g_i as follows, where $\text{nat}(x)$ is the natural width of the glue x , $\text{str}(x)$ is the infinite stretch factor of x , W is `\textwidth`, and n is the number of columns given to `\begin{paracol}`.

$$\begin{aligned}
 W' &= \sum_{i=0}^{n-2} (\text{nat}(\hat{w}_i) + \text{nat}(\hat{g}_i)) + \text{nat}(\hat{w}_{n-1}) \\
 F &= \sum_{i=0}^{n-2} (\text{str}(\hat{g}_i) + \text{str}(\hat{g}_i)) + \text{str}(\hat{w}_{n-1}) \\
 x_i &= \begin{cases} (W/W')\text{nat}(\hat{x}_i) & W' \geq W \vee F \leq 0 \\ \text{nat}(\hat{x}_i) + (\text{str}(\hat{x}_i)/F)(W - W') & W' < W \wedge F > 0 \end{cases} \quad (x \in \{w, g\})
 \end{aligned}$$

That is, if the total of natural widths W' is larger than `\textwidth` W or there are no infinite stretch factors in the specification, given widths are scaled down or up so that the scaled total is equal to W . Otherwise, each width with an infinite stretch factor is extended according to its ratio in the total stretch so that the stretched total is equal to W .

For a `paracol` environment with parallel-paging, n is replaced with n_l for the columns in left parallel-pages, while n , w_i and g_i are replaced with n_r , w_{n_r+i} and g_{n_r+i} for those in right parallel-pages. Moreover, if the optional argument having $s'_0, s'_1, \dots, s'_{k'}$ is provided, w_{n_r+i} and g_{n_r+i} for a column in right parallel-pages are determined by s'_i instead of s_i .

¹⁷If the command is in a `paracol` environment, the command does not affect the column widths of the environment but does the next ones, though such usage is very unusual.

¹⁸Or second most to not using it at all, of course.

- In `paracol` environments having n columns, s_i s.t. $i \geq n$ and \hat{g}_{n-1} are ignored. On the other hand if $k < n - 1$, it is assumed s_i is an empty string for all $i > k$.
- Finite stretch factors and finite or infinite shrink factors in \hat{w}_i and \hat{g}_i are ignored.
- Unlike \TeX 's genuine glue addition, all infinite unit `fil`, `fill` and `filll` are not distinguished in the summation for F . Also unlike \TeX 's genuine scaling of a glue primitive, $f\text{\fill}$ means `0pt plus f fill` for convenience¹⁹.
- The division W/W' and $str(\hat{x}_i)/F$ can have some arithmetic errors and thus the total of w_i and g_i may not be equal to W exactly but can be a little bit less than W . This small error is, however, equally distributed to g_i in typesetting of a page to make the total width of columns and gaps is exactly W ²⁰.
- All the specifications shown in the table below give us same results for a `paracol` environment having three columns, providing `\textwidth = 360pt` and `\columnsep = S = 20pt`.

s_0, s_1, s_2	w_0	g_0	w_1	g_1	w_2 (in pt)
<code>50pt/20pt, 100pt/40pt, 150pt</code>	50	20	100	40	150
<code>50pt, 100pt/2\columnsep, 150pt</code>	50	S	100	$2S$	150
<code>50pt/\fill, 100pt/2\fill, 150pt</code> <code>, 2\fill/2\columnsep, 3\fill</code>	50	$(1/3) \cdot 60$	100	$(2/3) \cdot 60$	150
<code>50pt/20, 50pt plus 1fil/40pt, 50pt plus 2fil</code> <code>5pt/2pt, 10pt/4pt, 15pt</code>	$(1/6) \cdot 300$	S	$(2/6) \cdot 300$	$2S$	$(3/6) \cdot 300$
<code>100pt/40pt, 200pt/80pt, 300pt</code>	50	20	$50 + (1/3) \cdot 150$	40	$50 + (2/3) \cdot 150$
	$10 \cdot 5$	$10 \cdot 2$	$10 \cdot 10$	$10 \cdot 4$	$10 \cdot 15$
	$0.5 \cdot 100$	$0.5 \cdot 40$	$0.5 \cdot 200$	$0.5 \cdot 80$	$0.5 \cdot 300$

- If your document has both of `\columnratio` and `\setcolumnwidth` prior to a `paracol` environment, the command given later is effective for the environment.

7.4 Commands for Two-Sided Typesetting and Marginal Note Placement

`\twosided[t_1 t_2 \dots t_k]`

The command enables a set of two-sided typesetting features $\{t_i \mid t_i \in \{\text{p, c, m, b}\}, 1 \leq i \leq k\}$ explicitly by the optional argument, or all of the following four features as a whole without the argument, in even-numbered pages.

`p(age)` for ordinary two-sided paging, letting the left side margin be `\evensidemargin`, page headers be different from those in odd-numbered pages with `headings` or `myheadings` page style, and `\cleardoublepage` leave an even-numbered page blank if it is used in an odd-numbered page.

`c(olumn)` for *column-swapping to print* columns in even-numbered pages in reverse order. This feature is sometimes preferable in typesetting especially with unbalanced parallel columns to make, for example, a wider columns are always *inside* while narrower ones are *outside*.

`m(arginal text)` to place marginal notes in the side margin opposite to that specified by the command `\marginparthreshold` discussed shortly.

`b(ackground painting)` to make background painting, shown in Section 7.8, *mirrored* so that, for example, a color specified for the left margin is used to paint the right margin instead.

- The feature `p` is also enabled by the `twoside` option of `\documentclass` with almost all classes including `article`, `book`, `report`, etc. Though it is strongly recommended to make both settings by `\documentclass` and this command consistent, they can be inconsistent resulting in lack of

¹⁹In \TeX 's grammar, $f\text{\fill}$ means a dimension rather than a glue and is `0pt` because the natural component of `\filll` is 0.

²⁰If we may ignore the arithmetic error inherent in \TeX .

some expected functions. For example, enabling `p` feature by `\twosided` without `twoside` option in `\documentclass` makes the format of headers and footers in all pages same even with `\pagestyle{headings}`.

- The column-swapping enabled by the feature `c` is ineffective in non-paired parallel-paging because it is meaningless²¹, and thus silently ignored.
- In ordinary single-column typesetting, marginal note swapping in even-numbered pages is enabled by the `twoside` option, while it never takes place in ordinary two-column typesetting. For marginal notes given in `paracol` environments, however, swapping of them in even-numbered pages is enabled by giving the feature `m` to `\twosided`.
- The command has to be outside of `paracol` environments to decide the action in the environments following them. If it appears in a `paracol` environment, you will have a warning message saying it is ignored.
- *This narrower, outside and italicized column-1 is at first in right side but the page break has changed the position to the left.*
- Here is an example of column swapping. Since this page 22 is odd, this wider column-0 with roman font is placed in left side and thus inside at the beginning, but now we are in an even page in which this column is in right side.
- In old versions of `paracol`, namely 1.2 and its minor revisions 1.2x, column-swapping was controlled by lengthy commmands `\swapcolumninevenpages` and `\noswapcolumninevenpages`. Though they are still available and will be so forever for backward compatibility, it is recommended to use `\twosided` with or without the feature `c`. The old versions also have a problem that spanning stuff crossing a page boundary is placed incorrectly after the page break in it, but this problem is solved by a fix incorporated in version 1.3.
- It must be $t_i \in \{\text{p, c, m, b}\}$, or you will have an error message of illegal two-siding feature.
- Section 9 shows examples of two-sided typesetting together with related issues on parallel-paging.

`\marginparthreshold{k}[k']`

The command specifies the minimum ordinal k of columns whose marginal notes are placed in right margin. That is, marginal notes given in a column- i go to left margin if $i < k$, while they go to right if $i \geq k$. The optional argument k' , if given, is for columns in right parallel-pages to decide the margin where their marginal notes are placed. In default, $k = 1$ is assumed to let marginal notes from the leftmost column-0 go to left margin while those from other columns go to right.

- You may specify $k = 0$ to let all marginal notes go to right margin, or may give the command a large number, say 100, to place all of them in left margin.
- The setting $k = 0$ or $k = 100$ above makes a side margin *shared* by marginal notes from different columns, and sharing is inevitable when a (parallel-) page has three or more columns. When a margin is shared by marginal notes from two or more columns, it can happen that two marginal notes from different columns conflict over the space to be occupied by each of them. This conflict is solved by `paracol` to push down the note given later in your source `.tex` until an available space for it is found. Note that the marginal note to be pushed down is determined by the position in the source rather than that in the printed result. Also note that `paracol` exploits space between two marginal notes having been already placed in the placement of other note coming later to place it at the natural position if possible or to minimize the amount of pushing down otherwise.
- In the decision of the real margin in which a marginal note is placed, other two factors are involved; `m` feature of `\twosided` command and the parity of the page; and L^AT_EX's genuine command `\reversemarginpar`. More specifically, after the first preliminary decision is made according to

²¹Unless somebody tells the author it is meaningful.

the threshold given to `\marginparthreshold`, we have the following two steps to modify the decision; if `m` feature has been specified in `\twosided` command and the marginal note belongs to an even-numbered page, the decision is reversed to have the second preliminary result; and then if `\reversemarginpar` has been specified, the second result is reversed (again) to have the final result.

- In old versions of `paracol`, namely older than 1.3, marginal note placement was not only uncontrollable but also gave ugly results when your document has three or more columns because the marginal notes from a column not being leftmost or rightmost were placed in the gap following the column rather than a margin. This miserable *gap note* placement does not happen any more, or in other words this is no more available because the author believes nobody loves it.
- Section 9 shows examples of marginal note placement together with related issues on parallel-paging and two-sided typesetting.

`\marginnote` [*left*] {*right*} [*voffset*]

You may use the package `marginnote` and its command `\marginnote` in `paracol` environments as a replacement of `\marginpar`. However, the command is *emulated* with `\marginpar` and `paracol`'s own mechanism of marginal note placement. Therefore, some of `marginnote`'s functionality are not effective in `paracol` environment except for the following features.

- Shifting up/down a marginal note by the optional *voffset*.
- Defining fonts (and others) for marginal notes by `\marginfont`.
- Controlling the horizontal paragraph alignment by `\raggedleftmarginnote` and `\raggedrightmarginnote`.

Note that you will see a warning message “`\marginnote` is emulated by `\marginpar`” at the first in-`paracol` occurrence of the command to let you know the imperfection.

7.5 Commands for Counters

`\globalcounter`{*ctr*}

`\globalcounter`*

The command `\globalcounter`{*ctr*} declares that the counter *ctr* is global to all columns, while `\globalcounter`* does so for all counters. An update of a global counter in a column is seen by any other columns.

- All column-local values of a descendant local counter of a global counter are zero-cleared when the global counter is explicitly stepped by `\stepcounter` or `\refstepcounter`, or implicitly by a sectioning command and so on.
- The counter `page` is always global but an explicit update of it by e.g., `\setcounter` in a non-leftmost column is not seen by other columns and is canceled even for the column itself after a column-switching or a page break in the column. Therefore, if you want to make a *jump* of `page`, it must be done in the leftmost column 0. Note that a jump from a page *p* to *q* can be seen in other columns even if they have gone beyond *p* before the column 0 makes the jump, as far as `page` having *q* (or its successor) is referred to by `\pageref` or through *contents* files such as `.toc`²².
- All counters except for `page` are local by default. This feature may cause a problem with some packages including `marginnote` and (auto-)p`st`-pdf having their own counters which must be global. Since it is tough to find the name of such counters from package sources, if you have something wrong with these (or other) packages, try to put `\globalcounter`* in your preamble and use `\localcounter` shown below to localize specific counters which you need to be local.

²²Direct reference to `page` may give an inconsistent result, as you might have in ordinary L^AT_EX documents.

- Globalizing a *ctr* being already global is just ignored without any complaints.

`\localcounter{ctr}`

The command declares that the counter *ctr* is local for each column.

- Though this command is intended for localizing a *ctr* which is once globalized, localizing a local counter does not causes any error but is just ignored. Localizing the permanently global `page` is also just ignored without any complaints.

`\definethecounter{ctr}{col}{rep}`

The command defines `\thectr` being *{rep}* for the local use in the column *col*. That is, `\thectr` in the column *col* acts as if it is defined by `\renewcommand{\thectr}{rep}`.

`\synccounter{ctr}`

The command *broadcasts* the value of the local counter *ctr* in the column in which the command appears to the values in all other columns.

`\syncallcounters`

The command broadcasts the values of all local counters in the column in which the command appears to the values in all other columns.

7.6 Page-Wise Footnotes

`\footnoteplacement{layout}`

`\footnotelayout{layout}`²³

The command specifies the *layout* $\in \{c, p, m\}$ of footnotes in `paracol` environments as follows.

c(column) makes footnotes *column-wise* (aka multi-columned) being default to place footnotes in each column at the bottom of the column and separating them from pre-environment and post-environment footnotes.

p(age) makes footnotes *page-wise* (aka single-columned) so that footnotes in all columns are gathered, typeset spanning all columns, and placed at the bottom of the page in which they appear or at the end of the `paracol` environment they belong to, so that they are separated from pre-environment and post-environment footnotes.

m(erge) makes page-wise footnotes *merged* with footnotes in outside of the environment but in the same page, i.e., those in pre-environment and post-environment stuff.

- An example of merged footnote is found in p. 15 while you will see many of them in Section 8²⁴.
- In any layouts, a footnote cannot have page breaks in it, i.e., a footnote is always put in a page as a whole. This makes it impossible to have a footnote taller than `\textheight` and thus you will see a warning message if you give a very long footnote which will be printed intruding into the area for page footer (or out of the paper bound).
- Choosing the layout page-wise or merged makes `footnote` counter global and `\fncounteradjustment` shown below performed inside `\footnoteplacement`. Choosing column-wise let the command do the operations oppositely, i.e., localizes `footnote` and does `\nofncounteradjustment`. Though these settings are usually appropriate for each footnote layout but you can override them by explicitly using commands like `\localcounter{footnote}`.

²⁴The left-column footnote 6 in p. 12 looks like a merged footnote because it is at the bottom of the page and the marked text is above the single-column text. However, it is an ordinary column-wise one produced by a trick with `\footnotemark` and `\footnotetext` in different `paracol` environments.

- The command has to be outside of `paracol` environments to decide the action in the environments following them. If it appears in a `paracol` environment, you will have a warning message saying it is ignored.
- In old versions of `paracol`, namely 1.2 and its minor revisions 1.2x, footnote layout was controlled by a set of commands `\multicolumnfootnotes` for `c`, `\singlecolumnfootnotes` for `p`, and `\mergedfootnotes` for `m`. Though they are still available and will be so forever for backward compatibility, it is recommended to use `\footnoteplacement`²⁵.
- It must be *layout* $\in \{c, p, m\}$, or you will have an error message of illegal layout specifier.

```
\footnote*[num]{text}
\footnotemark*[num]
\footnotetext*[num]{text}
```

The starred version of `\footnote`, `\footnotemark` and `\footnotetext` are for the adjustment of the footnote numbering, the order of footnote marks in main texts, and the stacking order of footnotes at page bottom. Their usages with various examples are given in Section 8.

```
\fncounteradjustment
\nofncounteradjustment
```

The maintenance of `footnote` with the starred footnote commands such as `\footnote*` shown above causes out-of-order progress of the counter to make it hard to have a consistent counter value at `\end{paracol}`. The command `\fncounteradjustment` is to let `\end{paracol}` adjust the value of the counter based on its value at `\begin{paracol}` and the number of footnote commands in the environment. The command `\nofncounteradjustment` is to tell `\end{paracol}` to do nothing as in default.

- Though `\footnoteplacement` with `p`(age-wise) or `m`(erged) argument does `\fncounteradjustment` while that with `c`(olumn) does `\nofncounteradjustment` inside of it, you can override these settings by explicitly putting a counter adjustment command after `\footnoteplacement`.
- The effect of `\fncounteradjustment` is shown in Section 8.

```
\belowfootnoteskip
```

The typesetting parameter specifies the amount of the space inserted below footnotes of single-column pre-environment stuff if it does not have bottom floats. The default amount is 0 pt, i.e., no space is added.

7.7 Commands for Coloring Texts and Column-Separating Rules

```
\columncolor[mode]{color}[col]
\normalcolumncolor[col]
```

The command `\columncolor` declares that the *default color* of a column is *color* or what it specifies by the combination with the optional *mode*. The command `\normalcolumncolor` declares the default color is what `\normalcolor` specifies, i.e., black usually. The target column of these commands is that in which the commands reside, or *col* if it specified.

- The command may be outside of `paracol` environment. If so and *col* is not provided, the target column is the leftmost 0.
- The default color declaration is *global*. Therefore, even if the command appears in a `paracol` environment (and even in some grouping structure in it), the declaration will be kept effective after `\end{paracol}` to determine the default color of the specified column in succeeding `paracol` environments.

²⁵Not only for the sake of it, but also for being familiar with this command which could have some advanced feature, for example to put gathered footnotes into a specific column, someday.

- To give a color to texts (and maybe other stuff) in a column correctly, you need to load `color` package or its relative (e.g., `xcolor`) which the implementation of coloring in `paracol` relies on.
- Coloring with `\color[mode]{color}` and other coloring commands in `paracol` environments is of course allowed. One caution is that the `\color` decides the color for following texts until other specification is given or the group surrounding the command is closed. Therefore, `\switchcolumn` does not affect the coloring but a color given to the texts in a column is also applied to the texts in the column to be switched to. This irrelativeness of coloring and column-switching is shown in the example below.

This column is colored blue because

```
\columncolor{blue}
```

is specified. Here we have a `\switchcolumn`.

The color of this paragraph is green because we are still in the environment of green coloring, which we are now closing.

Since the coloring environment has been closed, the color of this paragraph is the default blue. Now we have yet another and the last `\switchcolumn` to the right.

This column is colored red because

```
\columncolor{red}
```

is specified.

Now the color of the right column is changed to green because

```
\begin{color}{green}
```

is given prior to this paragraph. Now we have another `\switchcolumn` to go back to the left.

Since this paragraph is outside of the coloring environment, its color is the default red.

The default coloring of columns does not affect anything outside of `paracol` environment of course, and thus this sentence is not colored²⁶.

```
\coloredwordhyphenated
```

```
\nocoloredwordhyphenated
```

The command `\coloredwordhyphenated` allows the first word following a coloring command such as `\color` to be hyphenated, but at the same time make it possible that a line is broken before the word. The command `\nocoloredwordhyphenated` acts oppositely and thus line breaking before the first word and hyphenating it are inhibited. By default, `\coloredwordhyphenated` is effective.

- The implementation of `color` package and its relatives makes it impossible that *word* is hyphenated when it appears like `{\color{red}word ...}` or `\textcolor{word ...}`. This inhibition of the hyphenation is sometimes annoying especially when the document is multi-columned and thus a line is narrow and a column is written in a language having long words such as German. Therefore in `paracol` package, a trick is used to allow the *word* is hyphenated. However this trick being insertion of a null horizontal space has a side effect that the word can have a line break before it. Though this line break is usually unharmed, in a special occasion the break is undesirable and inappropriate by making it possible that the *half-colored* word ‘inappropriate’ is broken between ‘in’ and ‘appropriate’ without hyphenation. Therefore, if you find such a inappropriate break, use `\nocoloredwordhyphenated` as follows, for example.

```
{\nocoloredwordhyphenated in\textcolor{red}{appropriate}}
```

```
\colseprulecolor[mode]{color}[col]
```

```
\normalcolseprulecolor[col]
```

The command `\colseprulecolor` declares the color for *column-separating rules*, being the vertical rules drawn at the center of gaps between columns, is *color* or what it specifies by the combination with the optional *mode*. The command `\normalcolseprulecolor` declares the color of rules is what `\normalcolor` specifies, i.e., black usually. If the optional argument *col* is given, these commands specifies the color of the rule in the gap following the column whose ordinal is *col*, rather than all rules.

²⁶Or colored black as `\normalcolor` specifies.

- The rules are drawn if L^AT_EX's typesetting parameter `\columnseprule` for the rule width has non-zero value, e.g., `0.4pt` to obey the standard rule thickness. The rules are *not* drawn on page-wise stuff, i.e., pre-environment and post-environment stuff, page-wise floats or (merged or non-merged) page-wise footnotes of course but also spanning texts. Therefore, if a page has spanning texts, the rules are *broken* by them as shown in the red rule example below.

This is a left column paragraph preceding a spanning text. Of course the rule separating this and the next column starts from the top of this paragraph.

This is a right column paragraph preceding a spanning text given by the `\switchcolumn*` at its end.

An Example of Spanning Text Given by `\subsubsection*` Command

Since we have a spanning text above, the red rule separating this and the next column is broken by the text.

It is also natural that the rule separating this and the previous column is terminated at the end of this `paracol` environment.

- To give a color to rules correctly, you need to load `color` package or its relative (e.g., `xcolor`) which the implementation of coloring in `paracol` relies on.
- Once you give a color to rules in a specific gap with the optional `col`, another `\colseprulecolor` or `\normalcolseprulecolor` without `col` does *not* change the color of the rule in the gap.

7.8 Commands for Background Painting

```
\backgroundcolor{region}[mode]{color}
\backgroundcolor{region(x_0,y_0)}[mode]{color}
\backgroundcolor{region(x_0,y_0)(x_1,y_1)}[mode]{color}
```

The command declares that *background painting* of *region* is performed with *color* or what it specifies by the combination of the optional *mode*. The *region* whose background is painted is one of the following.

`c(olumn)` for all columns, or particular one if *region* is `c[col]` to specify its ordinal *col*.

`g(ap)` for all gaps between columns, or particular one if *region* is `g[col]` to specify the ordinal *col* of the column preceding the gap.

`s(panning)` for spanning texts.

`f(loat)` for page-wise floats.

`n(ote)` for (merged or non-merged) page-wise footnotes.

`p(re/post)` for pre-environment and post-environment stuff.

`t(op)` for top margin.

`b(ottom)` for bottom margin.

`l(ef)` for left margin.

`r(ight)` for right margin.

In addition, capitals of the keys above, i.e., `C`, `G`, `...`, `L`, are also legitimate for *under painting*. For example, you may specify to paint the background of a region, say top margin, by two `\backgroundcolor` with `t` and `T` and with different color arranging the size of the region of either `t` or `T` (or both of them) by the *extension* option shown below.

The optional (x_0, y_0) is to enlarge the region to be painted shifting its left-top and right-bottom corner outside by the dimension x_0 horizontally and y_0 vertically, or to shrink it with negative dimensions. This *extension* can be asymmetric giving another optional (x_1, y_1) so that it acts on the right-bottom corner while let (x_0, y_0) shift only the left-top corner. Moreover, you may make each extension *infinite* by giving 10000 pt (about 3.5 m) to x_0 , y_0 , x_1 and/or y_1 so that the corresponding region edge is shifted to the paper edge. Furthermore, this *infinite extension* can be terminated at the point α inside the corresponding paper edge by giving $10000\text{pt} - \alpha$ ($\alpha \leq 1000\text{pt}$) to an extension parameter x_0 , etc.

- A region whose color is not specified is not painted and thus left blank (or kept as painted by `\pagecolor` if you specify it).
- Under-painting of columns and gaps by `C` and `G` is made for regions different from those over-painting `c` and `g`. That is, under-painting is done ignoring all page-wise stuff and thus the height of the regions is always `\textheight + \maxdepth`. On the other hand, over-painting is only for chunks shrunk or separated by page-wise stuff.
- You may exploit the following painting order, where x_i is the i -th spanning text ($x \in \{\mathbf{s}, \mathbf{S}\}$) or i -th chunk followed by the i -th spanning text, m and n is the number of spanning texts and columns in a page respectively, to overlay a preceding region with a succeeding region, if your *printer* allows overlaid color painting.

$$\begin{aligned} & \mathbf{T} \rightarrow \mathbf{B} \rightarrow \mathbf{L} \rightarrow \mathbf{R} \rightarrow \mathbf{G}[0] \rightarrow \cdots \rightarrow \mathbf{G}[n-1] \rightarrow \mathbf{C}[0] \rightarrow \cdots \rightarrow \mathbf{C}[n-1] \\ & \rightarrow \mathbf{t} \rightarrow \mathbf{b} \rightarrow \mathbf{l} \rightarrow \mathbf{r} \rightarrow \mathbf{N} \rightarrow \mathbf{n} \rightarrow \{\mathbf{F}, \mathbf{P}\} \rightarrow \{\mathbf{f}, \mathbf{p}\} \rightarrow \mathbf{S}_1 \rightarrow \cdots \rightarrow \mathbf{S}_m \\ & \rightarrow \mathbf{g}_1[0] \rightarrow \cdots \mathbf{g}_1[n-2] \rightarrow \mathbf{c}_1[0] \rightarrow \cdots \mathbf{c}_1[n-1] \rightarrow \mathbf{s}_1 \\ & \rightarrow \cdots \\ & \rightarrow \mathbf{g}_m[0] \rightarrow \cdots \mathbf{g}_m[n-2] \rightarrow \mathbf{c}_m[0] \rightarrow \cdots \mathbf{c}_m[n-1] \rightarrow \mathbf{s}_m \\ & \rightarrow \mathbf{g}_{m+1}[0] \rightarrow \cdots \mathbf{g}_{m+1}[n-2] \rightarrow \mathbf{c}_{m+1}[0] \rightarrow \cdots \mathbf{c}_{m+1}[n-1] \end{aligned}$$

- If you specify `b` feature by `\twosided`, background painting is *mirrored* in even-numbered pages so that `l` and `L` mean right margin, `r` and `R` mean left margin, and asymmetric extensions are applied to right-top and left-bottom corners.
- To give a color for background painting correctly, you need to load `color` package or its relative (e.g., `xcolor`) which the implementation of coloring in `paracol` relies on.
- To paint margins and regions having infinite extension correctly, the parameters `\paperwidth` and `\paperheight` should be set properly by, for example, a paper selection option of `\documentclass`.
- Section 10 shows examples of background painting to give you more intuitive explanations of `\backgroundcolor` and its region specifications.

`\nobackgroundcolor{region}`
`\resetbackgroundcolor`

The command `\nobackgroundcolor` declares that the background of *region* is not painted, where *region* is one of legitimate region specifiers of `\backgroundcolor`. The command `\resetbackgroundcolor` declares no regions are painted and thus gives you the default state.

- If you specified the background painting of `c[col]` or `g[col]` by `\backgroundcolor`, the painting is *not* canceled by `\nobackgroundcolor` with `c` or `g` but without `[col]`. Similarly, once you made declarations of background painting of both `c` and `c[col]` (resp. `g` and `g[col]`), `\nobackgroundcolor` with `c[col]` (resp. `g[col]`) cancels the painting of `c[col]` (resp. `g[col]`) but the region will still be painted by the color you gave to `c` (resp. `g`).

`\pagemargin`

This is a (kind of) *length command*²⁷ to have the width of the *rim* area placed at each paper edge to inhibit background painting in the area. That is, the inner edges of the area are considered as virtual paper edges to block painting of all margins and regions having infinite extension to the edges, for example in order to avoid printing troubles caused by painting the rim area too close to the real paper edges. The default value of `\pagemargin` is 0 to allow paint anywhere in a paper.

7.9 Control of Contents Output

`\addcontentsonly{file}{col}`

The command inhibits the output of contents information to $file \in \{\text{toc}, \text{lof}, \text{lot}\}$ from columns other than col .

- For example, this manual has `\addcontentsonly{toc}{0}` to inhibit the contents information output from `\subsection` commands in the right column in Section 4 and 5, or the table should have duplicated entries of sub-sections.
- It must be $file \in \{\text{toc}, \text{lof}, \text{lot}\}$, or you will have an error message of illegal type of contents file.

7.10 Page Flushing Commands

`\flushpage`

The command flushes pages up to the *top page* in which the leading column resides. Deferred floats which can be put in the pages up to the top page are also flushed.

`\clearpage`

The command does what `\flushpage` does and then flushes all floats still deferred if any. The deferred float flushing beyond the top page takes place at first for column-wise ones creating float columns for them, and then for page-wise ones creating *float pages* only with page-wise floats, as L^AT_EX's `\clearpage` does outside `paracol` environment.

`\cleardoublepage`

The command does what L^AT_EX's `\cleardoublepage` does outside `paracol`. That is, it does `\clearpage` always and then leaves a blank page if it is even-numbered and two-sided `p(age)` feature is enabled by `twoside` option of `\documentclass` or `paracol`'s own `\twosided` command shown in Section 7.4.

- This command is equivalent to `\clearpage` in `paracol` environments for non-paired parallel-paging because `\clearpage` flushes *both* left and right parallel-pages.

²⁷In reality, it is a `\dimen` register rather than a `\skip` register.

8 Numbering and Placement of Page-Wise Footnotes

Here we have a simple example of page-wise but not-merged footnotes²⁸.

²⁸Because of the non-merged typesetting, this footnote is put above the example.

First left-column paragraph	First right-column paragraph
..... with a footnote ²⁹ with a footnote ³¹
Second left-column paragraph	Second right-column paragraph
..... with a footnote ³⁰ with a footnote ³²

²⁹First left-column footnote.

³⁰Second left-column footnote.

³¹First right-column footnote.

³²Second right-column footnote. This and all other footnotes above are page-wise and, since footnote typesetting is non-merged, they are put above the post-environment stuff.

As shown above, it is easy to have a reasonable result of footnote numbering and placement as far as your `paracol` environment is completely included in a page and you accept the numbering in left-column-first manner constructing the environment as follows exploiting the fact `footnote` is made global, where b is the value of `footnote` counter at `\begin{paracol}`, i.e., the number given to the footnote just preceding the environment, and thus $b = 28$ in the example above.

```
\begin{paracol}{2}
left-column stuff having n footnotes numbered b + 1, b + 2, ..., b + n
\switchcolumn
right-column stuff having m footnotes numbered b + n + 1, b + n + 2, ..., b + n + m
\end{paracol}
```

The real life is, however, tougher than that, because the assumptions above are too optimistic as described in the following subsections.

8.1 Multiple `\switchcolumn` in a Page

Here we have an example with three `\switchcolumn` commands in a page having six footnotes. Hereafter, footnotes are typeset with `\footnoteplacement{m}`³³.

First left-column paragraph	First right-column paragraph
..... with a footnote ³⁴ with a footnote ³⁶
Second left-column paragraph	It is followed by a <code>\switchcolumn*</code> .
..... with a footnote ³⁵	
It is followed by a <code>\switchcolumn</code> .	
Third and synchronized left-column paragraph .	Second and synchronized right-column paragraph
..... with a footnote ³⁷ with a footnote ³⁸
It is followed by a <code>\switchcolumn</code> .	Third right-column paragraph
 with a footnote ³⁹

³³And thus this footnote is merged with those in the `paracol` environment.

³⁴First left-column footnote.

³⁵Second left-column footnote.

³⁶First right-column footnote but following the second left-column one.

³⁷Third left-column footnote but following the first right-column one.

³⁸Second right-column footnote but following the third left-column one.

³⁹Third right-column footnote.

The example in the previous page should look weird because the order of the third footnote in the left column 37 and the first in the right 36 are reversed in their numbers and in the stack at the page bottom. However, the result is *natural* because they are numbered and stacked in the order of occurrence in the source .tex as always done in any documents without paracol and with it but column-wise footnote typesetting. Since the paracol cannot maintain the order automatically⁴⁰, you have to maintain it by yourself.

The problem is partly solved by using \footnote with its optional argument [num] to number the first right-column and the third left-column footnotes explicitly, i.e., to give num = 37 to the former and num = 36 to the latter. One caution is that you have to remember that \footnote with the optional num does not update footnote counter and thus you have to do \setcounter{footnote}{37} or \addtocounter{footnote}{2} after the third left-column footnote.

This remedy, however, cannot change the stacking order of these two footnotes of course. Therefore, you need another trick with \footnotemark and \footnotetext to stack the third left-column footnote above the first right-column one. More specifically, you can solve the problem inserting

```
\footnotetext[36]{text for the third left footnote}
```

somewhere between \footnote commands for the second left-column and the first right-column ones, e.g., at the end of the second left-column paragraph, and attaching its mark to the appropriate word for the footnote by \footnotemark[36], to have the following.

<p>First left-column paragraph with a footnote⁴¹ in it. Second left-column paragraph with a footnote⁴² in it. It is followed by \footnotetext[43]{text} and a \switchcolumn.</p> <p>Third and synchronized left-column paragraph with a footnote whose mark here⁴³ is given by \footnotemark[43] in it. It is followed by \addtocounter{footnote}{2} and a \switchcolumn.</p>	<p>First right-column paragraph..... with a footnote⁴⁴ in it. It is followed by a \switchcolumn*.</p> <p>Second and synchronized right-column paragraph with a footnote⁴⁵ in it. Third right-column paragraph..... with a footnote⁴⁶ in it.</p>
---	---

Though this solution gives a good result, however, it has the following two problems. First, you have to explicitly specify the footnote number through the optional arguments [num] of \footnote, \footnotetext and \footnotemark. This problem is quite severe because, for example, if you add a footnote somewhere preceding the paracol environment in question, you have to modify all [num] arguments of footnote-related commands in the environment. This means that when the footnote addition is done in the first page of a 100-page document having paracol environments with explicitly numbered footnotes in every page, you have to make the corrections for environments in 99 pages. The other a little bit less severe problem is that you have to keep footnote counter having correct value by \setcounter, \addtocounter or \stepcounter for footnotes following those with explicit numbering so that their numbers are given by the default action of \footnote.

To cope with these two problems, paracol provides you with the *starred* versions of \footnote and its relatives as introduced in Section 7.6 and detailedly explained in the next Section 8.2.

⁴⁰So far, because the maintenance is extremely tough. But since it is not impossible, some day you could have an improved version of paracol with the automatic ordering.

⁴¹First left-column footnote.

⁴²Second left-column footnote.

⁴³Third left-column footnote given by \footnotetext[43]{text} placed at the end of the second left-column paragraph.

⁴⁴First right-column footnote whose number 44 is explicitly given by \footnote[44]{text}.

⁴⁵Second right-column footnote correctly following the first right-column one.

⁴⁶Third right-column footnote.

8.2 Commands `\footnote*` and Relatives

```
\footnote* $[+disp]$ {text}
\footnote* $[-disp]$ {text}
\footnote* $[disp]$ {text}
```

The command is similar to its non-starred counterpart but the explicit numbering with the optional argument is done in *self-relative* or *base-displacement* style. That is, if the optional argument has a leading ‘+’ or ‘-’, the number given to the footnote is $f + disp$ or $f - disp$ respectively where f is the value of `footnote` counter, or in other words the number given to the last footnote⁴⁷. Otherwise, i.e., the optional argument is a number without +/- sign, the number given to the footnote is $b + disp$ where b is the base value of `footnote` counter at `\begin{paracol}` for the environment in which the command appears, or in other words the number given to the last pre-environment footnote⁴⁸.

In addition, unlike the non-starred version, this command updates `footnote` counter with the number given to the footnote, i.e., $f \leftarrow f + disp$, $f \leftarrow f - disp$ or $f \leftarrow b + disp$ is performed, so that following `\footnote` without explicit numbering option have numbers $f + 1$, $f + 2$ and so on with new f .

- If the optional argument is not provided, it is assumed that `[+1]` is given and thus `\footnote*{text}` acts as `\footnote{text}`.

```
\footnotemark* $[+/-]disp$ 
```

This command is a mixture of its non-starred counterpart and `\footnote*`. That is the number for the footnote mark is calculated in the way of `\footnote*` and `footnote` counter is updated.

```
\footnotetext* $[+/-]disp$ {text}
```

Without the optional argument `[+/-]disp`, this command does what `\footnotetext{text}` does but in addition increments `footnote` counter before that. With the optional argument, on the other hand, the number given to the footnote `text` is calculated as done in `\footnote`, but the `footnote` counter is not updated.

With these starred commands, you can produce the following using the base-displacement mechanism without worrying about the absolute value of `\footnote` counter and its change.

<p>First left-column paragraph with a footnote⁴⁹ in it. Second left-column paragraph with a footnote⁵⁰ in it. It is followed by <code>\footnotetext*[3]{text}</code> and a <code>\switchcolumn</code>. Third and synchronized left-column paragraph with a footnote whose mark here⁵¹ is given by <code>\footnotemark*[3]</code> because $51 = 48 + 3$. It is followed by a <code>\switchcolumn</code>.</p>	<p>First right-column paragraph with a footnote⁵² in it. It is followed by a <code>\switchcolumn*</code>. Second and synchronized right-column paragraph with a footnote⁵³ in it. Third right-column paragraph with a footnote⁵⁴ in it.</p>
---	--

⁴⁷If it is put by the ordinary `\footnote`.

⁴⁸Or the last footnote in the previous `paracol` environment, etc.

⁴⁹First left-column footnote.

⁵⁰Second left-column footnote.

⁵¹Third left-column footnote given by `\footnotetext*[3]{text}` placed at the end of the second left-column paragraph to have $51 = 48 + 3$.

⁵²First right-column footnote whose number 52 is given by `\footnote*[4]{text}` because $52 = 48 + 4$.

⁵³Second right-column footnote produced by `\footnote*[5]{text}` because $53 = 48 + 5$.

⁵⁴Third right-column footnote produced by `\footnote{text}` because $54 = 53 + 1$.

The other way to produce the same result except for the absolute footnote numbers is to use the self-relative mechanism and to exploit the progress of `footnote` counter as follows.

<p>First left-column paragraph with a footnote⁵⁵ in it. Second left-column paragraph with a footnote⁵⁶ in it. It is followed by <code>\footnotetext*{text}</code> and a <code>\switchcolumn</code>.</p> <p>Third and synchronized left-column paragraph with a footnote whose mark here⁵⁷ is given by <code>\footnotemark*[-1]</code> because $57 = 58 - 1$. It is followed by a <code>\switchcolumn</code>.</p>	<p>First right-column paragraph with a footnote⁵⁸ in it. It is followed by a <code>\switchcolumn*</code>.</p> <p>Second and synchronized right-column paragraph with a footnote⁵⁹ in it. Third right-column paragraph with a footnote⁶⁰ in it.</p>
--	--

It depends on the structure of your document which of the base-displacement and self-relative is better. If your document has frequent switching between single- and multi-column text typesetting and thus the contents of a `paracol` environment is relatively small, the base-displacement is a good choice because you may concentrate on one base value of `footnote` counter. Otherwise, especially when your document consists of one single and large `paracol` environment, the base-displacement is almost equivalent to maintaining absolute values and thus the self-relative should be preferred.

Note that if the last `\footnote` or `\footnotemark` in a `paracol` environment is starred, the command lets `footnote` counter have some value smaller than that for the last stacked footnote. For example, if the second and third right-column footnotes `59` and `60` are omitted from the example above, the last footnote-related command will be `\footnotemark*[-1]` which makes the counter at `\end{paracol}` `57` rather than `58`. You may not worry about this problem, however, because `\end{paracol}` automatically maintains the counter letting it have $b + n$ where n is the number of `\footnote` and `\footnotemark` in the environment, if the maintenance is ordered by the command `\fncounteradjustment` which is automatically executed by `\footnoteplacement` with the argument `p` or `m`.

8.3 Page Break

When a `paracol` environment with footnotes lays across a page boundary, you could have some weird result even if the environment have just one `\switchcolumn` as shown below.

<p>First left-column paragraph with a footnote⁶¹ in it.</p>	<p>First right-column paragraph with a footnote⁶³ in it.</p>
--	--

⁵⁵First left-column footnote.
⁵⁶Second left-column footnote.
⁵⁷Third left-column footnote given by `\footnotetext*{text}` placed at the end of the second left-column paragraph because it follows the second footnote `56`.
⁵⁸First right-column footnote whose number `58` is given by `\footnote{text}` because $58 = 57 + 1$ and `\footnotetext*` for `57` lets `footnote` have the value.
⁵⁹Second right-column footnote produced by `\footnote*{+2}{text}` because $59 = 57 + 2$.
⁶⁰Third right-column footnote produced by `\footnote{text}` because $60 = 59 + 1$.
⁶¹First left-column footnote.

Second left-column paragraph with a footnote ⁶⁹ in it. It is also followed by a <code>\switchcolumn</code> .	Second right-column paragraph with a footnote ⁷⁰ in it.
---	---

Unfortunately, this tactics does not always solve the problem. If a left-column paragraph has a page break in it and a footnote before the break, doing `\switchcolumn` after the paragraph is too late to let right-column footnotes reside in the page just having been broken, while inserting `\switchcolumn` before the paragraph should cause incorrect stacking order.

The remedy for this problem is similar to that shown in Section 8.1 to cope with multiple `\switchcolumn` in a `paracol` environment. Here it is shown a little bit more formally. Suppose we have a page in a `paracol` environment in which a page break occurs in p_l -th and p_r -th paragraphs in the left and right columns respectively. Thus we have $p_l - 1$ and $p_r - 1$ completed paragraphs in each of both columns. Let n_l (resp. n_r) be the number of footnotes in the pre-break left-column (resp. right-column) paragraphs, and m_l (resp. m_r) be the number of pre-break footnotes in the p_l -th (resp. p_r -th) paragraph. Thus we have $n_l + m_l$ (resp. $n_r + m_r$) footnotes in the left (resp. right) column of the page before the break. The following construct assures that those footnotes are properly numbered and stacked at the bottom of the page.

```

First to  $(p_l - 1)$ -th paragraphs with  $n_l$  footnotes in total given by \footnote{text}.
\footnotetext*{1st footnote in  $p_l$ -th paragraph}
...
\footnotetext*{ $m_l$ -th footnote in  $p_l$ -th paragraph}
\switchcolumn
First to  $(p_r - 1)$ -th paragraphs with  $n_r$  footnotes in total given by \footnote{text}.
\footnotetext*{1st footnote in  $p_r$ -th paragraph}
...
\footnotetext*{ $m_r$ -th footnote in  $p_r$ -th paragraph}
\switchcolumn
 $p_l$ -th paragraph whose first footnote mark is given by \footnotemark*[-( $m_l+n_r+m_r-1$ )], while
second to  $m_l$ -th ones are given by \footnotemark without * nor optional [num]. The first
subsequent footnotes beyond the page break, if any, is given by \footnote*[( $n_r+m_r+1$ )]{text}
while further subsequent ones are given by \footnote{text}.
\switchcolumn
 $p_r$ -th paragraph whose first footnote mark is given by \footnotemark*[-( $m_r+k_l-1$ )] where  $k_l$  is
the number of left-column footnotes beyond the break, while second to  $m_r$ -th ones are given
by \footnotemark. The first subsequent footnotes beyond the page break, if any, is given by
\footnote*[( $k_l+1$ )]{text}, while further subsequent ones are given by \footnote{text}.

```

The example shown in the next two pages is for the case of $p_l = p_r = n_l = n_r = m_l = m_r = k_l = 2$.

⁶⁹Second left-column footnote whose number 69 follows the right-column footnote 68 in the last page.

⁷⁰Second right-column footnote whose number 70 follows the left-column footnote 69.

...and two post-break footnotes.....	...and two post-break footnotes.....
... here ⁷⁹ by <code>\footnote*{+5}{text}</code> here ⁸¹ by <code>\footnote*{+3}{text}</code>
...and here ⁸⁰ by <code>\footnote{text}</code>and here ⁸² by <code>\footnote{text}</code>

followed by a `\switchcolumn`.

Note that though the remedy works well as shown above, it is not a good idea to do that when you are writing draft versions of your document because page break points go up and down by your modifications to the document. Therefore, it is recommended to put all footnotes by non-starred `\footnote` until your document becomes perfect except for footnote numbering and placement and then to adjust them by the technique described in this section.

⁷⁹Fifth left-column footnote given by `\footnote*{+5}` because $n_r + m_r + 1 = 2 + 2 + 1 = 5$ and thus $79 = 74 + 5$.

⁸⁰Sixth left-column footnote given by `\footnote{text}`.

⁸¹Fifth right-column footnote given by `\footnote*{+3}` because $k_l + 1 = 3$ and thus $81 = 78 + 3$.

⁸²Sixth right-column footnote given by `\footnote{text}`.

page 39 (now), this wider column is now left one keeping it inside, while the marginal note given in the first line of this page goes to right and outside. Now we will have a `\switchcolumn` below this paragraph to go to the column-1 and back to the previous page 38.

by the page break. The third marginal note Third is given in the first line of this page, but marginal it is pushed down again due to the conflict note from with the note from the column-0. column-0

Note that the position of the last marginal note in the `paracol` environment which we just have closed *Third marginal note from column-1. Marginal* affects the marginal note placement in post-environment stuff. For example, the marginal note given in the first line of this paragraph is pushed down.

We will see a few examples of parallel-paging shortly, but before that we will have an intentional black note given page to make the first page of the example odd-numbered to avoid you have an impression that its layout is after incorrect⁸⁵ because if it were in an even page you would see the *outside* third and fourth supplementary *columns* `paracol` environment is closed.

⁸⁵At least the author himself had such impression without the blank page.

(intentionally blanked page)

This is the first paragraph of the column-2 being the left column of the right parallel-page. Though we are in a page different from that column-0 and 1 reside in, this page is still numbered 41 because the left and right page is paired. Therefore, the left margin of this page is narrower than the right margin because the page number is odd.

You have to notice the first paragraph does not start from the page top but above it we have some space of exactly same size as the pre-environment stuff shown in the left parallel-page. Therefore, the top of the first paragraphs in all columns are aligned. The marginal note given in the first line of this paragraph goes to the right margin of this page because of the `\marginparthreshold` setting and the parity of this page. Now we have a `\switchcolumn` to the next column-3.

We have a few other materials not shown in right parallel-pages. The space above this paragraph is for the spanning text placed in the left parallel-page. The page-wise footnote given here⁸⁷ is also not in this page but in the left. Finally, the author has put a page-wise figure spanning columns just before `\switchcolumn` by which we left this column, but it will be in the right page 42 together with column-0 and 1.

.....

Though the footnote numbered 87 goes to the left page, its space and that of 86 make this and the next

This is the first paragraph in the last rightmost Marginal column-3 whose width is equal to that of the column- note from 2. The marginal note given in the first line goes column-3. to right and does not conflict with that from the column-2. We are now going back to the column-0 by a `\switchcolumn` with a spanning text.*

Marginal note from column-2.

As expected, this line is aligned to the first line of the paragraph in the column-2 as well as those in column-0 and 1. It is also consistent the first lines including that of this paragraph are not indented because the spanning text is given by `\subsection` which makes first paragraphs unindented.*

.....

After the page break we will have shortly, this column becomes the leftmost in the left parallel-page,

Another marginal note from column-3. as you are seeing now, but still outermost as well as the marginal note in the outside left margin.

columns shorter in the previous page. Similarly, we have a space above for the page-wise figure shown in the right page.

page-wise figure given in column-2

Figure 11: A Page-Wise Figure

Another marginal note from column-1. column-0 and is placed outside (left) in the page, as well as the marginal note in this right page but in the outside margin.

the break but in the right one. This is because the feature `c` is enabled to swap not only columns in a page but also the left and right paired parallel-pages when they are even-numbered. The other feature `p` makes the left outside margins of this right and the previous left pages wider than the right inside margins.

Now you are seeing yet another material placed only in the page in which the column-0 resides and thus being the right page now, i.e., this paragraph and the next one in the post-environment stuff. You might be disappointed by the fact the *outside* pages, i.e., left in this page 42 and right in the previous page 41, cannot have page-wise stuff but it is what the author can do now for the version 1.3 and thus you have to wait some future versions in which the author could devise a mechanism to exploit the corresponding space in the pages⁸⁸. In addition, you might think it is weird that the `c` feature of `\twosided` swaps columns *and* paired pages. However this swapping is a natural consequence of the combination of column-swapping and paired parallel-paging. Therefore, you can simply disable the `c` feature (maybe together with other features) to have more intuitive results.

In the next Section 9.2, you will see another kind of parallel-paging namely non-paired one. Before that, we need a blank page to let the non-paired parallel-paging start from an even-numbered page so that a left and right page pair comprises a double spread. A short remark on the blank next page is that it does not have a right counterpart parallel-page because the page is outside `paracol` environments and does not have any portion from the environments⁸⁹.

⁸⁸You might complain the immaturity of parallel-paging and might claim that it should be included in `paracol` after the author implements the mechanism. In fact the author himself is frustrated current features of parallel-paging but he dared to release the version 1.3 knowing that there are people who happily typeset their parallel-paged documents with the current limited features.

⁸⁹To illustrate this fact, the author dares to put a real blank page rather than stepping the `page` counter.

(intentionally blanked page)

9.2 Example of Non-Paired Parallel-Paging

This and following three pages are to show an example of non-paired parallel-paging, in which the author keeps the setting of `\twosided`, `\columnratio` and `\marginparthreshold` unchanged. The arguments of `\begin{paracol}` for column population are also unchanged to have 2 + 2 configuration, but the first argument is followed by `*` for non-paired typesetting. That is, the environment below starts by `\begin{paracol}[2]*{4}`. The contents of the environment is also almost same as the previous Section 9.1, while **bold-faced** words show the difference from the paired typesetting.

Marginal note from column-0.

This is the first paragraph of the leftmost column-0, whose first line has a marginal note placed in the **left** margin because the setting of `\marginparthreshold` being 0 is still effective and we are in the **even**-numbered page 44. Now we have a `\switchcolumn` to the next column-1.

Marginal note from column-1.

This is the first paragraph of the second and right column-1 in the left parallel-page. We shortly give an italicized marginal note carefully, so that it does not conflict with the marginal note from the column-0. That is, now the author puts the note. Now we have a `\switchcolumn` to the next column-2.

A Spanning Text: though this is wider than the page width, this text does not span the boundary between the left and right parallel-pages.

We have come back to this column-0. The space above the spanning text is due to the synchronization because two paragraphs in the column-2 are significantly taller in total than the paragraphs in other columns. As the spanning text itself says, it cannot extend to the right parallel-page. The author puts dummy lines to go to the page bottom.

.....

We have restarted this column-1. This paragraph has a footnote⁹⁰ as shown below.

.....

Now we will have a page break shortly. You **will not** be surprised by seeing this column is still in the left parallel-

After the page break below, this column also stays in the left page together with

⁹⁰This footnote is put in the left parallel-page together with another footnote below given in the column-2 in the right parallel-page.
⁹¹This footnote is *not* put in the right parallel-page though it is given in the column-2 in the right parallel-page and thus its reference is in the column, of course.

This is the first paragraph of the column-2 being the left column of the right parallel-page. **Since we are in the page next to** that column-0 and 1 reside in, this page is numbered **45** because the left and right page is **non-paired**. Therefore, the left margin of this page is narrower than the right margin because the page number is odd.

You have to notice the first paragraph does not start from the page top but above it we have some space of exactly same size as the pre-environment stuff shown in the left parallel-page. Therefore, the top of the first paragraphs in all columns are aligned. The marginal note given in the first line of this paragraph goes to the right margin of this page because of the `\marginparthreshold` setting and the parity of this page. Now we have a `\switchcolumn` to the next column-3.

We have a few other materials not shown in right parallel-pages. The space above this paragraph is for the spanning text placed in the left parallel-page. The page-wise footnote given here⁹¹ is also not in this page but in the left. Finally, the author has put a page-wise figure spanning columns just before `\switchcolumn` by which we left this column, but it will be in the **left** page **46** together with column-0 and 1.

.....

Though the footnote numbered **91** goes to the left page, its space and that of **90** make this and the next

This is the first paragraph in the last rightmost Marginal column-3 whose width is equal to that of the column- note from 2. The marginal note given in the first line goes column-3. to right and does not conflict with that from the column-2. We are now going back to the column-0 by a `\switchcolumn` with a spanning text.*

Marginal note from column-2.

As expected, this line is aligned to the first line of the paragraph in the column-2 as well as those in column-0 and 1. It is also consistent the first lines including that of this paragraph are not indented because the spanning text is given by `\subsection` which makes first paragraphs unindented.*

.....

After the page break we will have shortly, this column is kept being the rightmost in the right

page-wise figure given in column-2

Figure 12: A Page-Wise Figure

*Another
marginal
note from
column-1.*

page after the break. This is because the feature `c` is **not effective in non-paired parallel-paging**. The other feature `p` consistently makes the left outside margins of this and the previous page in which this column resides wider than the right inside margins.

*the column-0 and is placed **inside (right)** in the page, as well as the marginal note in this **left page still** in the outside margin.*

As the post-environment stuff in Section 9.1 is, this paragraph being the post-environment stuff of the non-paired parallel-pages appears only in the parallel-page in which the column-0 belongs to, and thus in the left parallel-page in this case.

columns shorter in the previous page. Similarly, we have a space above for the page-wise figure shown in the **left** page.

parallel-page, as you are seeing now, **and** still out- Another
ermost as well as the marginal note in the outside marginal
right margin. note from
column-3.

10 Examples of Background Painting

10.1 Fundamental Painting

As you undoubtedly notice, this page and a few pages following it are colorfully painted. For this and the next three pages, the author declared the background color of each region as follows.

```
\backgroundcolor{t}[rgb]{0.7,0,0}      % dark red for top margin
\backgroundcolor{b}[rgb]{0.8,0.6,0}    % dark orange for bottom margin
\backgroundcolor{l}[rgb]{0,0,0.7}      % dark blue for left margin
\backgroundcolor{r}[rgb]{0,0.7,0}      % dark green for right margin
\backgroundcolor{c[0]}[rgb]{1,0.8,1}   % pink for column-0
\backgroundcolor{c[1]}[rgb]{1,1,0.8}   % cream yellow for column-1
\backgroundcolor{g}[rgb]{0.8,1,1}      % light blue for the gap
\backgroundcolor{f}[rgb]{0.8,0,1}      % purple for page-wise floats
\backgroundcolor{n}[rgb]{0.8,0.6,1}    % light purple for page-wise footnotes
\backgroundcolor{p}[rgb]{0.8,1,0.6}    % pale green for pre/post-environment
\backgroundcolor{s}[rgb]{0.8,0.8,0.8}  % light gray for spanning texts
```

Therefore, the background of this pre-environment paragraph and other stuff above is painted by pale green. Since the author set `\pagemargin` to be 5 pt, you will see unpainted strips of 5 pt wide at all paper edges surrounding painted regions. For this and the next three pages, `\twosided[pcm]` is declared to enable p, c and m features but to disable the b feature. Therefore, though this page 48 is even and thus the left outside margin is wider than the right inside one, the backgrounds of l(ef) and r(igh) margins are painted by dark blue and dark green respectively.

As explained in the right column-0, the background of this left and outside column-1 is painted by cream yellow as `\backgroundcolor{c[1]}` specifies. Now we have a `\switchcolumn` with a spanning text to show the background painting for it⁹².*

This column-0 is now right and inside because of the c feature of `\twosided` is enabled. On the other hand, the background of this column is painted by pink because `\backgroundcolor` for c[0] specifies so. That is, the column ordinals optionally given to c(olumn) (and g(ap)) regions are logical ones not always corresponding to their *physical* positions in a page.

The background of this s(panning text) region is painted by light gray

See the right column for the reason why this paragraph is here.

This paragraph is to show how the first line of a paragraph just below a spanning text is placed in the painted region.

See the right column for what we are now doing.

Now we have a `\flushpage` to see the background painting for a material not shown in the page, i.e., a page-wise float.

⁹²Since the footnotes in this `paracol` environment are page-wise and merged, and `\backgroundcolor{n}` specifies light purple, the background of this (foot)n(ote) region is painted by the color.

f(float) region for this page-wise figure is painted by purple

Figure 13: A Page-Wise Figure

Since we are now in an odd-numbered page 49, this column-0 is now a left one and is still painted by pink of course.

As expected, the background of this column-1 is still painted by cream yellow.

This paragraph is to show how the last line of a page without page-wise footnotes is placed in the painted region.

See the comment in the left column.

See the right column for the reason why we have this almost blank page.

This page is to show how the page without any page-wise stuff looks like.

See the right column for what will happen shortly.

Shortly we will close this `paracol` environment in the next page.

Now we are closing this `paracol` environment to show how its post-environment stuff is painted.

See the left column for the reason why we are now closing the environment.

The background of this paragraph in `p(ost-environment)` region is also painted by pale green, because post-environment stuff can be pre-environment stuff at the same time as we see shortly.

This short `paracol` environment illustrates how the pre-environment stuff of this environment, or the post-environment stuff of the last environment in other words, is painted.

Therefore, the author does not have much to say in this column, except for giving a footnote here⁹³.

Before moving to the next example, one caution is given for background painting of merged footnotes. As the footnote [93](#) itself says, merged footnotes given in the last page of a `paracol` environment are considered as belonging to post-environment stuff. Therefore, the footnote [93](#) is painted by pale green as well as another footnote given now⁹⁴.

⁹³Since this footnote is merged with that in the post-environment stuff, it is considered as a part of post-environment stuff and thus painted by pale green rather than light purple.

⁹⁴Since this footnote really belongs to post-environment stuff, its background is painted by pale green naturally.

10.2 Mirrored Painting and Enlarging/Shrinking/Shifting Regions

At a glance, this and the next three pages look painted similarly to previous four pages, but by a careful examination you should notice two important differences. The first one is found in the colors of left and right margins. As the author enabled all features of `\twosided` including `b` for mirroring and we are now in an even-numbered page 52, the left and outside margin is painted by dark green for the region `r`(ight margin), while the right and inside one is painted by dark blue for `l`(eft margin).

The other is that regions are enlarged, shrunk or shifted by 4pt by the following `\backgroundcolor` commands with extensions.

```
\backgroundcolor{t(0pt,0pt)(0pt,-4pt)}[rgb]{0.7,0,0} % B up
\backgroundcolor{b(0pt,-4pt)(0pt,0pt)}[rgb]{0.8,0.6,0} % T down
\backgroundcolor{l(0pt,4pt)(-4pt,4pt)}[rgb]{0,0,0.7} % R left T/B outside
\backgroundcolor{r(-4pt,4pt)(0pt,4pt)}[rgb]{0,0.7,0} % L right T/B outside
\backgroundcolor{c[0](4pt,4pt)}[rgb]{1,0.8,1} % all edges outside
\backgroundcolor{c[1](4pt,4pt)}[rgb]{1,1,0.8} % all edges outside
\backgroundcolor{g(-4pt,4pt)}[rgb]{0.8,1,1} % L/R inside & T/B outside
\backgroundcolor{f(4pt,4pt)(4pt,-4pt)}[rgb]{0.8,0,1} % L/R outside & T/B up
\backgroundcolor{n(4pt,-4pt)(4pt,4pt)}[rgb]{0.8,0.6,1} % L/R outside & T/B down
\backgroundcolor{p(4pt,4pt)}[rgb]{0.8,1,0.6} % all edges outside
\backgroundcolor{s(4pt,-4pt)}[rgb]{0.8,0.8,0.8} % L/R outside & T/B inside
```

In the comments above, L(ef), R(ight), T(op) and B(ottom) mean edges moved by a given extension. Therefore, for example, “L/R outside & T/B up” for `f`(loat) region means it is enlarged horizontally and shifted up vertically by the asymmetric extension `(4pt,4pt)(4pt,-4pt)`. These a little bit complicated setting of extensions are to solve the problems in the fundamental example shown in previous four pages, namely too strict definition of the regions to be painted. That is, both vertical edges of regions having texts, e.g., `c`(olumn) regions, should look too close to the first and last letters. Similarly both horizontal edges of those regions seem too close especially when the first line is tall (e.g., the section title in p. 48 and the page-wise figure in p. 49) and the last line of a column is followed by spanning text or post-environment stuff. Therefore, the author made fine tuning moving inside edges of margins outside, and so on. We will come back this issue after exemplifying the effect of the tuning.

This paragraph is surrounded by spaces of a small but comfortable amount as well.
95.

By the tuning to enlarge this `c`(olumn) region, this paragraph has comfortable spaces above and below it, as well as at the both side edges.

The background of this `s`(panning text) region is painted by light gray and enlarged horizontally but shrunk vertically

See the right column for the reason why this paragraph is here.

This paragraph is to show how well the first line of a paragraph just below a spanning text is separated from the boundary of two painted regions.

See the right column for what we are now doing.

By enlarging this `c`(olumn) region and shift the (foot)n(ote) region down, this paragraph has a comfortable amount of space below it.

⁹⁵Shifting this (foot)n(ote) region down a little bit, the space below this footnote and above the top edge of the `b`(ottom margin) region is enlarged.

shifting up this f(loat) region gives us a small space above the top edge of the rectangle

Figure 14: A Page-Wise Figure

Similarly to other paragraphs below page-wise stuff, this paragraph is well separated from the bottom edge of the f(loat) region above.

See the comment in the left column for the intention of placing this paragraph here.

As in the case of the line above page-wise footnotes, the last line of this paragraph has a sufficient space separating it from the top edge of the b(ottom margin) region.

See the comment in the left column, too.

See the right column for the reason why we have this almost blank page.

This page is to show how the page without any page-wise stuff looks like. As you are seeing, the space above this paragraph is sufficient and comfortable.

See the right column for what will happen shortly.

Shortly we will close this `paracol` environment in the next page.

Now we are closing this `paracol` environment to show how this paragraph is separated from the boundary of `c(olumn)` and `p(ost-environment)` regions.

See the left column for the reason why we are now closing the environment.

The background of this paragraph in `p(ost-environment)` region is painted by pale green as done in p. 51, but its top and bottom edges *look* shifted down and up to give spaces below and above the last and first paragraphs in `paracol` environments, respectively.

This short `paracol` environment illustrates how the pre-environment stuff of this environment, or the post-environment stuff of the last environment in other words, is painted.

Therefore, the author does not have much to say in this column, except for giving a footnote here⁹⁶.

In the setting with `\backgroundcolor` commands in p. 52, the author carefully moved contacting edges of regions. For example, to enlarge `c(olumn)` regions, the inside edges of `l(eft margin)` and `r(ight margin)` regions are moved outside, and both vertical edges of the `g(ap)` region shifted toward its inside. As for the horizontal edges, the bottom edges of `t(op margin)` and `f(loat)` regions are moved up, the top edges of `b(ottom margin)` and `(foot)n(ote)` regions are moved down, and both top and bottom edges of the `s(panning text)` region are shifted toward its inside.

These edge shifting could make a region too narrow or too much shifted resulting in a material in it overreaching its boundary, especially in vertical shifting of horizontal edges. However we can exploit some large space automatically or manually inserted above and/or below the material to avoid overreaching. That is the author exploited the following spaces; `\headsep` below the page head (though it is empty in this document); `\dbltextfloatsep` below the bottom-most page-wise float; spaces that `\subsection*` inserts above and below it together with manually inserted `\medskip` below it; `\skip\footins`⁹⁷ above the first footnote which the author enlarged by 4pt temporarily for this and the next subsections; and `\footskip` from the bottom edge of text area to that of the page number.

Now you might notice that the explanation above does not mention the `p` region for pre-environment and post-environment stuff. As you should find in the settings, this region is enlarged horizontally *and vertically* so that its top and bottom edges are moved up and down when the region is at the top or bottom of a page, as you are seeing now and find in p. 52. However, this enlargement of course has a side effect that the region collides against `c(olumn)` and `g(ap)` regions also enlarged vertically making them overlapped. This overlap will be invisible with most of *printers* because, as shown in Section 7.8, `p` region is painted *before* `c` and `g` regions are painted. In addition, since relatively large spaces of `\bigskip` are manually inserted before each `\begin{paracol}` and after each `\end{paracol}`, texts in pre-environment and post-environment stuff are well separated from region boundaries.

This overlay painting `c` and `g` over `p`, however, might produce an unexpected result with some printer with which, for example, two colors are *blended* in the thin overlapped strip⁹⁸. Unfortunately, this overlay painting is inevitable in the current version 1.3, but in a future version, hopefully 1.4, more sophisticated *position-dependent* region definition, for example, to shift the top edge of `p` region only when the region is at the top of page, could be introduced.

Another remark is that the mirroring specified by the `b` feature of `\twosided` works not only on the colors of side margins but also on their asymmetric shrinkage. That is, the asymmetric shifts of vertical edges of `l` and `r` regions correctly performed irrespective of their physical positions, i.e., even when the `l` (resp. `r`) region is at the right (resp. left) margin and the edge to be shift is the left (resp. right) one rather than right (resp. left).

⁹⁶As the footnote 93 in p. 51, this merged footnote is a part of post-environment stuff and thus painted by pale green rather than light purple.

⁹⁷This is a kind of “length command” maybe not widely known.

⁹⁸For example, a dvi previewer `dviout` produces such a blended result with the default setting of coloring.

10.3 Regions with Infinite Extensions

You are now seeing another background painting much different from previous two examples. That is, after disabling painting of `t`, `b`, `l`, `r` and `g` regions by `\nobackgroundcolor`, the author gave the followings for painting this and the next pages.

```
\backgroundcolor{c[0](4pt,4pt)(0.5\columnsep,4pt)}[rgb]{1,0.8,1}
\backgroundcolor{c[1](0.5\columnsep,4pt)(4pt,4pt)}[rgb]{1,1,0.8}
\backgroundcolor{C[0](10000pt,10000pt)(0.5\columnsep,10000pt)}[rgb]{1,0.8,1}
\backgroundcolor{C[1](0.5\columnsep,10000pt)(10000pt,10000pt)}[rgb]{1,1,0.8}
```

The first two lines above is different from the previous declaration because inside edges of `c[0]` and `c[1]` regions are shifted toward outside of them and thus inside of unpainted `g` region so that the edges are contacted. On the other hand, the last two lines are for *under-painting* of columns and has *infinite extension* to make top, bottom and outside edges of `C` regions reaching to the corresponding paper edges. Since this under-painting is done with colors same as those of over-painting of `c` regions, you will have an impression that the paper is two-toned and page-wise stuff are pasted on the paper⁹⁹.

As explained in the right column, this `c[1]` region also has an invisible left edge shifted left by 4pt¹⁰⁰.

Though you cannot see, the right edge of this over-painted `c[0]` region is shifted right by 4pt to hide the small patch at the right bottom corner of the `p` region above by overlaying.

This s(panning text) region could be extended to both side edges of the paper if its extension were (10000pt,-4pt).

Little to say as well.

The author does not have much to say now for this column chunk.

Nothing to say as well.

Still nothing to say particular to the page break we will have shortly.

⁹⁹This footnote is given outside `paracol` environment but its background is painted by light purple because it is merged with the footnote 100.

¹⁰⁰This (foot)n(ote) region could be extended to both side edges and the bottom edge of the paper if its extension were (10000pt,-4pt)(10000pt,10000pt).

This figure is given in the `paracol` environment closed in the previous page but its background is not painted.

Figure 16: A Page-Wise Figure *Exported* to Post-Environment

This `f`(loat) region could be extended to both side edges and the top edge of the paper if its extension were $(10000\text{pt}, 10000\text{pt}) (10000\text{pt}, -4\text{pt})$.

Figure 15: A Page-Wise Figure *Imported* from Pre-Environment

This paragraph is just for keeping the `paracol` environment alive in this page.

This paragraph is not necessary for keeping alive the environment but is given for consistent view.

Note that overlay painting is inevitable for two-toned page painting, as far as you want to paint background of page-wise stuff.

The last issue of background painting is about painting materials given outside `paracol`. As you have seen, pre-environment and post-environment stuff are painted but it is done only when they reside in a page having a portion of a `paracol` environment (maybe) of course. Therefore, the next page is *not* painted because the page does not have any parallel-columned stuff. Therefore, even if you wish to paint the whole of your document including pages without `paracol` stuff, you cannot do it just with `paracol` package, at least so far.

On the other hand, some materials given outside `paracol` environments are painted as if they are given in the environment when they are *imported* into the environment. One category has footnotes given in pre-environment stuff when `\footnoteplacement{m}` is specified for merging, as exemplified by the footnote 99 in the previous page. Note that such a footnote is painted by the color for `n` region rather than `p` region even when there are no footnotes in the `paracol` environment. The other category has ordinary floats given by `figure` and/or `table` (i.e., neither `figure*` nor `table*`) environments outside `paracol` and then *deferred* to a page having (a portion of) stuff produced by `paracol`. Since such a float, e.g., Figure 15 in this page, is considered as a page-wise float given in the `paracol` environment in this section, its background is painted by the color for the `f` region, rather than that for the `p` region which would be used if the float were placed in the previous page. Note that such a deferred float import could occur not only from the page having `\begin{paracol}` but also from pages preceding it. For example, if you have three `figure` environments in a page $p - 1$ just preceding the page p in which you start a `paracol` environment, it could happen that first one is placed in $p - 1$ without painting, the second is placed in p and painted by the color for `p`, and the third is placed in $p + 1$ and painted by the color for `f`.

Finally some materials *exported* from a `paracol` environment are painted as if they are in post-environment stuff. In previous two subsections, we saw merged footnotes (e.g., 93 in p. 51 and 96 in p. 55) are painted by the color of `p` rather than `n`. The other kind of exportation is of page-wise floats given in a `paracol` environment but deferred to the page next to the page having `\end{paracol}`, or further. For example, Figure 16 is given in the `paracol` environment above in this page, but its background is not painted because the next page in which the figure is placed does not have any parallel-columned stuff¹⁰¹.

¹⁰¹If it has, the background is painted by the color for `p`.

(intentionally blanked page to show this page is *not* painted)

11 Known and Unknown Problems

Here a few problems you could face in the use of `paracol` are summarized.

- If your (e.g.,) left column goes ahead too much farther than the right column, \LaTeX could stop with the following error message.

```
! Package paracol Error: Too many unprocessed columns/floats.
```

This usually means that the internal space to keep materials in the left column is exhausted. More specifically, suppose at some point in your `.tex` the left column is in the page p while the right is in $q < p$. We need $(p - q)$ boxes to keep the left column contents in the pages $q, q + 1, \dots, p - 1$ because these pages cannot be *printed* yet until the right column fills them. In addition, we also need two boxes for the left column in p and the right column in q so that you make column-switching between them keeping unprinted contents in them. Therefore, at least we need to have $(p - q) + 2$ boxes, while the number of them provided by \LaTeX is only 18¹⁰². Therefore, `paracol` cannot continue its work if $(p - q)$ reaches 17. Furthermore, other stuff also consumes the boxes as follows.

- If there are n pages in $q, q + 1, \dots, p$ having pre-environment stuff or page-wise floats, n boxes are consumed by them. Similarly, if m pages in them have page-wise footnotes, m boxes are given to them.
- If the left (resp. right) column has column-wise footnotes in p (resp. q), a box is used for them.
- If the left (resp. right) column has k floats to be placed in p (resp. q) or to be deferred to $p + 1$ (resp. $q + 1$) or a succeeding page, k boxes are reserved for them.

Therefore, it should be safe to keep $(p - q)$ from exceeding 10 or so placing `\switchcolumn` in both columns fairly frequently.

- As discussed in Section 7.2, setting a synchronization point in a page brings the following side effects.
 - Stretch and shrink factors of all vertical skips in the page are nullified. The nullification of stretch factors could make a sparse column in the page have a vertical space at its bottom as if `\raggedbottom` setting is in effect even with `\flushbottom` one, rather than distributing the amount of the space to the skips so that the bottom line is aligned at the page bottom. As for the nullification of shrink factors, it makes the page have lines a little bit less than that it would have without synchronization because lines above the (last) synchronization point cannot be compressed. The other effect is a little bit subtle because the shrink factors below the last synchronization point are taken care of by \TeX 's page builder when it examine the appropriateness of each breakable point, but they are nullified when the page is printed. That is, if \TeX finds a good break point which needs that the stuff between the synchronization and break points is compressed a little bit, the stuff is printed without compression making its bottom edge a little bit below the page bottom.
 - After a synchronization point is set, columns in the page cannot have top floats any more even if a column has space above the synchronization point and large enough to place the float. Therefore, if you like to exploit the space, you have to place the `figure` or `table` environment in question prior to the column-switching command or environment for the synchronization.
- As the author did for Section 1 to 5, sometimes you will make a section header spanning all columns by giving a sectioning command such as `\section`, `\subsection` and `\subsubsection` to the optional

¹⁰²Readers who are acquainted with \LaTeX implementation will understand that 18 is the cardinality of the set $\{\backslash\text{bx@A}, \dots, \backslash\text{bx@R}\}$ for floats acquired by `\newinsert`. Those who are more familiar with that might know that most \LaTeX , based on e- \TeX or others having similar extensions, now have 52 `\inserts` $\{\backslash\text{bx@A}, \dots, \backslash\text{bx@Z}, \backslash\text{bx@AA}, \dots, \backslash\text{bx@ZZ}\}$ for floats and materials of `paracol`, since 2015

argument of `\switchcolumn*` or `\begin` of a synchronizing column-switching environment. These three commands work well and you will have what you intend to have, but you have to be careful with lower-level commands `\paragraph` and `\subparagraph`. Unlike higher-level relatives, these lower-level commands does *not* put the header *immediately* but keep it somewhere¹⁰³ so that when the paragraph following the command starts it is put as the leading part of the paragraph. Therefore if the spanning text has (e.g.) `\paragraph` only, the header is not put as a spanning text but at the head of the first paragraph of the column to which you switch, leaving an empty spanning text with some large space as follows.

This left-column paragraph precedes a synchronized column-switching.

This right-column paragraph precedes a synchronized column-switching.

A Spanning Text Given by `\paragraph` This left-column paragraph follows the synchronization but is led by `\paragraph` given to the optional argument of `\switchcolumn*` for spanning text.

This right-column paragraph follows the synchronization with an empty spanning text.

Therefore, unless this is what you intend to do, you have to give some paragraph together with `\paragraph` to the optional argument for spanning text. For example, `\mbox{}` is a good candidate as the paragraph following `\paragraph` because it produces (almost) nothing. By using this technique the example above becomes the followings.

This left-column paragraph precedes a synchronized column-switching.

This right-column paragraph precedes a synchronized column-switching.

A Spanning Text Given by `\paragraph` Followed by `\mbox{}`

This left-column paragraph follows the spanning text above.

This right-column paragraph follows the spanning text above.

-
- As shown in Section 8, it is not easy to have good numbering and stacking order of page-wise footnotes even with the supports from `\footnote*` and its relatives. In addition, a footnote in a `paracol` environment cannot be broken into two (or more) pages.
 - As the author confessed in Section 9.1, right parallel-pages cannot have page-wise stuff but have blank spaces in the corresponding region for them. The author will try to remove this limitation from a future version of `paracol`, in the version 1.4 hopefully.
 - As discussed in Section 10.2, it is desirable that background painting region definition in `\backgroundcolor` has position dependent extensions. The author is fairly optimistic about the incorporation of this advanced feature in the version 1.4.
 - In the release dated 2015/01/10, L^AT_EX changed its mechanism of the placement of double-column floats (or in our terminology, page-wise floats) to avoid *out-of-order* appearance of them. That is, until the release on 2014/05/01 a double-column float (e.g., `figure*`) can be overtaken by a single-column float of the same category (e.g., `figure`) when they cannot be put into the page in which texts around them are put. In order to cope with the problem, the new version merged two lists to keep *deferred* double- and single-column floats into one so that the appearance order of them is determined by their order in the single list. Though this change should have made people happy when they typeset *ordinary* two-column (or multiple-column) documents, the new feature might not be welcomed by `paracol` users because your

¹⁰³For people familiar to T_EX's *dangerous bends*, the header is kept in `\everypar`.

parallel-columns have their own *streams* of floats to be put in the corresponding columns. Therefore, and for the sake of simplicity of `paracol`'s implementation, the author decided to nullify this new feature in `paracol` environments. That is, even with new releases of L^AT_EX, your page-wise floats given in a `paracol` environment can be overtaken by column-wise floats.

In addition to the problems above known to the author, there may be (or should be, honestly speaking) other unknown problems in `paracol` because it cannot be perfect though the author has made his best effort for testing and debugging it. Particularly, sometimes it is very tough, if not impossible, to make `paracol` compatible with other packages, especially with those having dark magic as `paracol` has in it¹⁰⁴. Therefore, though reporting incompatibleness with a package you use is very welcome¹⁰⁵, you should kindly understand the toughness of the compatibility issue.

Furthermore, even without such problematic packages, `paracol` might produce weird results due to its bug. If your document has something to make unknown bugs visible, you might have one (or more) of the followings which the author encountered in his debugging work.

- A page, a column, a footnote and/or a float disappears¹⁰⁶.
- A page, a column, a footnote and/or a float is duplicated.
- A message like “Overfull \vbox (1.23456pt too high) has occurred while \ouptut is active” is shown.
- A message “Underfull \vbox (badness 10000) has occurred while \ouptut is active” is shown. This message, however, does not always mean a bug but may just be a complaint that a column or a page is too sparse to meet your request to align the bottom of all columns and pages by `\flushbottom` setting. Therefore, if you have this message and you cannot be sure whether it means a bug or not, try `\raggedbottom` setting to see if you still have the message, before sending a bug report to the author.

If you encounter anything like them (or whatever you cannot solve by yourself), don't hesitate to report it to the author with minimum source file to produce the problem¹⁰⁷.

¹⁰⁴For example, the author knows it is almost impossible to make `paracol` compatible with one of the author's own package available in CTAN.

¹⁰⁵For example, `paracol` is now compatible with `color` package thanks to a report from a user.

¹⁰⁶In fact, a bug fixed in version 1.2 caused page losing though it happens very very rarely but an unlucky user encountered it.

¹⁰⁷And with patience because your problem might not be solved quickly.

PART II

Implementation

1 Overview

1.1 Column-Pages

In our parallel multi-column typesetting, a column may grow independently of other columns and may cross its page boundary asynchronously with others. Therefore, we cannot throw away the contents of a column in a page, or a *column-page* in short, when a page break occurs in the column. Instead, we have to keep column-pages until all columns are *synchronized* implicitly or explicitly.

An *implicit synchronization* takes place when all columns in a page see page-breaks to let the page is shipped out. In general, all columns but the last one which arrives the page-break have completed column-pages in the page in question and some of them may have succeeding column-pages. Therefore, we maintain the list of completed column-pages $S_c = \backslash\text{pcol@shipped}\cdot c$ for each column $c \in [0, C)$, where $C = \backslash\text{pcol@ncol}$ is the number of columns given through the argument of `paracol` environment, and the set of them $S = \{S_c \mid c \in [0, C)\}$.

Each element $s_c(p)$ of a list S_c is an `\insert` whose `\vbox` contains the p -th completed column-page¹⁰⁸, where $p = 0$ for the first column-page produced in `paracol` environment or that following a page flushing macro `\flushpage`, `\clearpage` or `\cleardoublepage`. That is, S_c is defined as follows, where $p_b = \backslash\text{pcol@basepage}$ is the zero-origin ordinal of the *base page* being the oldest page not shipped out yet.

$$\begin{aligned} S_c &= (s_c(p_b), s_c(p_b+1), \dots, s_c(p_b+k-1)) \\ &= \backslash\@elt s_c(p_b) \backslash\@elt s_c(p_b+1) \cdots \backslash\@elt s_c(p_b+k-1) \end{aligned}$$

Note that a list S_c can be empty and all members in S may be empty.

The other type of synchronization, *explicit synchronization*, takes place by `\switchcolumn*` or the beginning of starred column-switching environments, by `\end{paracol}`, or by one of page flushing macros `\flushpage`, `\clearpage` and `\cleardoublepage`. A flushing explicit synchronization ships out the pages from p_b to $p_t = \backslash\text{pcol@toppage}$ being the ordinal of the *top page* to which the most advanced *leading column* has reached. On the other hand, other non-flushing explicit synchronization keeps the page p_t from being shipped out because the column-pages in it or the page itself will grow further.

1.2 Current Column-Pages and Their Contexts

We also have to maintain another type of column-pages which are currently built, or *current column-pages* in short, to switch from a column to another. Since each column may have its own *context* for the typesetting of it, or *column-context* in short, it were perfect to save the context when we leave from a column and to restore that when we revisit the column if we could. However, `TEX` and `LATEX` has a tremendously large number of context variables and the number becomes virtually boundless when we take variables defined in various styles and by users themselves into account. Therefore, we had to abandon to keep the whole context of the column but carefully chose a small subset comprising variables automatically modified outside

¹⁰⁸Other registers such as `\count` are not used.

of users' control. That is, the column-context $\kappa_c = \text{\pcol@col}\cdot c$ of a column c consists of the following elements, each of which named e is referred to as $\kappa_c(e)$ hereafter.

- β represents $\text{\insert}\cdot\beta$ containing the followings.
 - $\beta^b = \text{\box}\cdot\beta = \text{\@holdpg}$ is the \vbox containing the main vertical list which has already contributed to the current column-page.
 - $\beta^p = \text{\count}\cdot\beta = \text{\pcol@page}$ means the current column-page belongs to the page β^p .
 - $\beta^r = \text{\dimen}\cdot\beta = \text{\@colroom}$ is the room of the column.
- $\tau = \text{\pcol@currfoot}$ is the \insert containing the footnotes added in the current column-page, if column-wise footnote typesetting is in effect. Its constituent \box , \count , \dimen and \skip are denoted as τ^b , τ^c , τ^d and τ^s respectively. On the other hand, if page-wise footnote typesetting is in effect, τ is always empty¹⁰⁹.
- $\delta = \text{\pcol@prevdepth}$ is the depth of the last vertical item in β^b obtained by \prevdepth .
- $\lambda_t = \text{\@toplist}$ is the list of top floats inserted in the current column-page.
- $\lambda_m = \text{\@midlist}$ is the list of mid floats inserted in the current column-page.
- $\lambda_b = \text{\@botlist}$ is the list of bottom floats inserted in the current column-page.
- $\lambda_d = \text{\@deferlist}$ is the list of column-wise floats deferred to the next column-page.
- $\xi = \text{\pcol@textfloatsep}$ is the vertical skip used instead of \textfloatsep for top floats in the current column-page if it has synchronization points, or ∞ otherwise.
- $\eta = \text{\@textfloatsheight}$ is the total height of mid floats and their separators in the current column-page.
- $\nu_t = \text{\@topnum}$ is the maximum number of top floats which the current column-page can accommodate further.
- $\rho_t = \text{\@toproom}$ is the room for top floats in the current column-page.
- $\nu_b = \text{\@botnum}$ is the maximum number of bottom floats which the current column-page can accommodate further.
- $\rho_b = \text{\@botroom}$ is the room for bottom floats in the current column-page.
- $\nu_c = \text{\@colnum}$ is the maximum total number of floats which the current column-page can accommodate further.
- σ is the following encoding of \if@nobreak and \if@afterindent at the time we left from the column c .

$$\sigma = \begin{cases} 0 & \text{\if@nobreak} = \text{false} \\ 1 & \text{\if@nobreak} = \text{true} \wedge \text{\if@afterindent} = \text{true} \\ 2 & \text{\if@nobreak} = \text{true} \wedge \text{\if@afterindent} = \text{false} \end{cases}$$

Note that we have only three states because \if@afterindent is meaningful only when $\text{\if@nobreak} = \text{true}$ ¹¹⁰.

¹⁰⁹But the macro \pcol@currfoot is used to keep page-wise footnotes temporarily.

¹¹⁰If only with the standard L^AT_EX and so far.

- $\varepsilon = \backslash\text{everypar}$ is the tokens stored in $\backslash\text{everypar}$ at the time we left from the column c .

In addition, we have special context variables $w_c = \backslash\text{pcol@columnwidth}\cdot c$ in which we keep $\backslash\text{columnwidth}$ for the column c .

Note that we could add other variables to the saved context and/or provide some API macro to define them by users, but abandon them because it should be too complicated for users¹¹¹. Also note that we provide a save/restore mechanism for local counters as discussed in §1.4.

1.3 Pages and Their Contexts

Besides the column-pages, we have to keep track each whole page not yet shipped out but has some complete or incomplete (i.e., current) column-pages. We maintain the list;

$$\begin{aligned} \Pi &= \backslash\text{pcol@pages} = (\pi(p_b), \pi(p_b+1), \dots, \pi(p_t-1)) \\ &= \backslash\@elt \pi(p_b) \backslash\@elt \pi(p_b+1) \dots \backslash\@elt \pi(p_t-1) \\ \pi(p) &= \{\pi^p(p)\}\pi^i(p)\pi^f(p)\{\pi^s(p)\}\{\pi^m(p)\} \end{aligned}$$

where $\pi(p)$ is the *page context* of p and its elements $\pi^p(p)$, $\pi^i(p)$, $\pi^f(p)$, $\pi^s(p)$ and $\pi^m(p)$ have the followings.

- $\pi^p(p) = \text{page}(p)$ is the value of the counter `page` (i.e. $\backslash\text{c@page}$) for the page p .
- Iff $\pi^i(p) \neq \perp$, the page p has page-wise floats or the single-column *pre-environment stuff* preceding $\backslash\text{begin}\{\text{paracol}\}$ in the *starting page* where it resides and spanning all columns. In this case $\pi^i(p) = i$ represents $\backslash\text{insert}\cdot i$, often *cached* in the macro $\backslash\text{pcol@spanning}$, for such *spanning stuff* whose registers have the followings.
 - $\pi^b(p) = \backslash\text{box}\cdot i$ contains the spanning stuff.
 - $\pi^h(p) = \backslash\text{dimen}\cdot i = \backslash\@colht$ if positive for the height of columns shrunk by the spanning stuff. If negative, the page is only for the spanning stuff, i.e. a *float page*. We use the notation $\pi^h(p)$ for the pages $\pi^i(p) = \perp$ to mean $\backslash\text{textheight}$.
 - $\pi^t(p) = \backslash\text{skip}\cdot i = \backslash\text{pcol@topskip}$ being the value of $\backslash\text{topskip}$ at $\backslash\text{begin}\{\text{paracol}\}$ to be inserted at the top of each column in each non-first page. Otherwise, i.e., for the columns in the starting page following the pre-environment stuff, it has 0 to prevent the $\backslash\text{topskip}$ insertion. We use the notation $\pi^t(p)$ for the pages $\pi^i(p) = \perp$ to mean $\backslash\text{pcol@topskip}$.
- Iff $\pi^f(p) \neq \perp$, page-wise footnote typesetting, discussed in §1.5, is in effect and the page p has some footnotes in $\backslash\text{box}\cdot \pi^f(p)$. This element is often *cached* in the macro $\backslash\text{pcol@footins}$.
- $\pi^s(p) = (\text{span}(H_1, h_1), \dots, \text{span}(H_n, h_n)) = \backslash\@elt\{H_1, h_1\} \dots \backslash\@elt\{H_n, h_n\}$ is the list of spanning texts in the page p , where i -th one's top edge is at H_i from the top of the page (excluding spanning stuff) and its height-plus-depth is h_i , where H_i and h_i are represented in the form of integers. Therefore, it is emptied by $\backslash\text{pcol@startpage}$, and then the elements are added by $\backslash\text{pcol@makecol}$ (only for the last one) and $\backslash\text{pcol@output@switch}$ whenever they find a spanning text completes. The element is often *cached* in the macro $\backslash\text{pcol@sptextlist}$ and is referred to by $\backslash\text{pcol@buildcolseprule}$ to draw column-separating rule and to paint columns and column-separating gap leaving spaces for spanning texts. The usage of this element is discussed in §1.7 a little bit more detailedly.

¹¹¹And for the author if we include save/restore of macros, though it could be done with a $\backslash\text{toks}$ containing the $\backslash\text{definitions}$ of macros.

- $\pi^m(p) = \{M_L^l\}\{M_L^r\}\{M_R^l\}\{M_R^r\}$ is the set of lists of marginal notes in the left (l) and right (r) margins and in the left (L) and right (R) parallel-pages. The words left and right of margins mean physical left and right, while left and right of parallel-pages mean the logical ones, i.e., the page where the column-0 resides is left. Each element $M_{\{L,R\}}^{\{l,r\}}$ has a list $(mpar(t_1, b_1), \dots, mpar(t_n, b_n)) = \backslash\@elt\{t_1\}\{b_1\} \dots \backslash\@elt\{t_n\}\{b_n\}$ of marginal notes whose top and bottom are at t_i and b_i from the top of the column area, where t_i and b_i are represented in the form of integers. Each element can be empty of course, and $\pi^m(p)$ itself can be so as well to mean all elements are empty¹¹². Therefore, $\pi^m(p)$ is emptied by `\pcol@startpage`, and then examined and modified by `\pcol@addmarginpar` when it adds a marginal note through macros `\pcol@getmparbottom` and `\pcol@setmpbelt`. Another modifier `\pcol@output@start` initializes one of the element $M_L^{\{l,r\}}$ with the value representing the last marginal note in pre-environment stuff, while another examiner `\pcol@output@end` lets the outside `\@mparbottom` have a value based on b_n of one of the element, according to L^AT_EX's setting of marginal note placement. The whole element $\pi^m(p)$ is often *cached* in the macro `\pcol@mparbottom`. The usage of this element is discussed in §1.7 a little bit more detailedly.

Note that even in parallel-paging and in non-paired one in particular, a page p consists of all columns $c \in [0, C)$. Therefore, the term *left/right parallel-page p* always mean the left and right component of a parallel-page (pair) p .

The reason why we keep track of $page(p)$ is that page numbering is not necessary to be consecutive. If such a *jump* occurs randomly in any columns explicitly updating `page`, it is very tough to give a consistent view of the page number of a specific page to all columns. Therefore we suppose jumps occur only in the leftmost column 0¹¹³ which controls the page numbering, while non-leftmost columns are expected to refer the `page` passively.

This page numbering is implemented as follows. Each time a column-page at p of the leftmost column is completed to start a new column-page, $page(p)$ is fixed to be the value of `page` and $page(q) = \pi^p(q)$ for all $q \in [p, p_t]$ are let be $page(p) + (q - p)$ in usual cases but $page(p) + 2(q - p)$ in non-paired parallel-paging. This update also takes place on column-switching from the leftmost column-page at p to another column so that a jump happening before the switching is notified to other columns. On the other hand, starting or column-switching to a non-leftmost column-page at p lets `page` have $page(p)$ referring to $\pi(p)$, unless the column starts the most advanced top page. In this new top page case, $\pi(p_t+1)$ is added to Π with the temporary setting $\pi^p(p_t+1) = page(p_t+1) = page(p_t) + 1$ usually but $\pi^p(p_t+1) = page(p_t+1) = page(p_t) + 2$ in non-paired parallel-paging, and p_t is incremented.

Note that this management is imperfect because direct references of `page` in non-leftmost columns can give inconsistent results if `page` is modified in a non-leftmost column or the reference occurs in a page p after that the leftmost column modifies `page` in a page q such that $q \leq p$. In addition to them, this mechanism in non-paired parallel-paging always gives incorrect page number to the columns in a right parallel-page because $\pi^p(p)$ always has $page(p)$ for the left parallel-page. However, it is expected that the progress of the leftmost column usually precedes other columns to give consistent `page` reference even with jumps, unless the reference is made by a column in a right non-paired parallel-page. More importantly, it is assured that indirect references through `.aux` records and page numbers recorded in `.toc`, `.idx`, and so on are always consistent because of the lazy evaluation of `page = page(p)` at ship-out of an ordinary page p or a left parallel-page p , while the counter is let have $page(p) + 1$ when a right

¹¹²To minimize the possibility of miscoding for emptying and save a small amount of memory for pages having no marginal notes.

¹¹³But we neither inhibit nor nullify a jump in non-leftmost column and thus the update can be seen referring to `page` counter explicitly.

non-paired parallel-page p is shipped out.

Also note that we also keep $\pi(p_t)$ in `\pcol@currpage` which is initialized by `\pcol@output@start` to let $\pi^i(p_t)$ have the pre-environment stuff. Then the macro is redefined to have the value representing the new page possibly with $\pi^i(p_t)$ for page-wise floats in `\pcol@startpage` by the macro `\pcol@defcurrpage`. Another `\definition` is done in `\pcol@output@switch` also with `\pcol@defcurrpage` to let $\pi^f(p_t)$ have page-wise footnotes built in `\footins` if page-wise typesetting is in effect and the column-switching leaves the column in p_t ¹¹⁴. We denote the concatenation of Π and $\pi(p_t)$ as Π^+ to represent all pages *on-the-fly*.

1.4 Counters

Besides the context variables discussed in §1.2, we need to make counters local to each column except for those declared to be global by `\globalcounter`. Let Θ be the set (list) of all counters declared before `\begin{paracol}`, i.e., $\Theta = \text{\clocckpt}$, and

$$\Theta^g = \text{\pcol@gcounters} = \{\theta_1^g, \dots\} = \text{\@elt}\{\theta_1^g\} \dots$$

be the set of *global counters* which have declared so by `\globalcounter{\theta_i^g}`. Then the set of *local counters* Θ^l is defined as follows.

$$\Theta^l = \Theta - \Theta^g = \text{\pcol@counters} = \{\theta_1^l, \dots\} = \text{\@elt}\{\theta_1^l\} \dots$$

Since each column has its own values in local counters, we have to keep the set of counter/value pairs

$$\Theta_c = \text{\pcol@counters}\cdot c = \{\langle \theta_1^l, \text{val}_c(\theta_1^l) \rangle, \dots\} = \text{\@elt}\{\theta_1^l\}\{\text{val}_c(\theta_1^l)\} \dots$$

for each column c , where $\text{val}_c(\theta_i^l)$ is the value of a counter θ_i^l local to c . That is, whenever we switch from a column c to d , we save $\langle \theta_i^l, \text{val}_c(\theta_i^l) \rangle$ in Θ_c and restore θ_i^l for d by letting it have $\text{val}_d(\theta_i^l)$ in Θ_d , for all $\theta_i^l \in \Theta^l$.

A global counter is free from these save/restore operations but needs another special operation when it is incremented by `\stepcounter`. That is, the invocation of `\stepcounter` for a global counter θ_i^g may clear local counters in its set of descendant counters $\zeta(\theta_i^g) = \text{\pcol@cl@}\theta_i^g$ and this clearing must be performed on the all instances of $\theta_j^l \in \zeta(\theta_i^g)$ saved in Θ_c for all $c \in [0, C)$. Therefore, on the `\stepcounter`, we do the followings for all $c \in [0, C)$; temporarily restore all $\theta_k^l \in \Theta^l$ from Θ_c ; clear all $\theta_j^l \in \zeta(\theta_i^g)$; and then save $\langle \theta_k^l, \text{val}_c(\theta_k^l) \rangle$ back to Θ_c .

The other item we maintain for a local counter θ^l is its *local representation* $\langle \text{rep} \rangle$ in a column c defined by `\definethecounter{\theta^l}\langle c \rangle \langle \text{rep} \rangle`. The local representation $\langle \text{rep} \rangle$ is kept in `\pcol@thectr@}\theta^l}\cdot c` and is made `\let`-equal to `\the}\theta^l` when the column c is visited.

1.5 Page-Wise and Merged Footnotes

Page-wise footnote typesetting is completely different from ordinary *column-wise footnote* typesetting.

When a column-page in the top page is built, `\footins` keeps all footnotes `\inserted` by `\footnote` or `\footnotetext` in *any* columns in the page. Therefore, `\footnote` and `\footnotetext` in the top page act as usual to add the footnote to `\footins`. Then if a column-switching takes place to leave the column, `\footins` is saved into $\pi^f(p_t)$ by `\pcol@`

¹¹⁴The `\definition` of `\pcol@currpage` in `\pcol@setpnoelt`, and emptying it in `\pcol@output@start` and `\pcol@freshpage` are for coding trick and thus not for giving a really new `\definitions`.

`output@switch`, so that $\pi^f(p_t)$ is inserted to `\footins` again by `\pcol@restartcolumn` when it visits a column in p_t , or by `\pcol@startcolumn` when it finds a column proceeds to p_t .

Then, when a column-page in the top page completes advancing p_t , `\footins` is kept in $\pi^f(p_t-1)$ by `\pcol@startpage`, rather than being combined with the column-page. This saving into $\pi^f(p_t-1)$ fixes the footnotes in p_t-1 so that $\pi^f(p_t-1)$ is combined with other materials in the page by `\pcol@outputelt` or `\pcol@makeflushedpage` through `\pcol@putfootins` when the page is shipped out.

Fixing $\pi^f(p)$ for $p < p_t$ makes it impossible to add footnotes in a column in the page p not only to $\pi^f(p)$ but also to `\footins` for the page p because we have at least one fixed column-page $s_c(p)$ unable to shrink to have such additional footnotes in p ¹¹⁵. Therefore, such a footnote addition is *deferred* and is thrown into $\pi^f(p_t)$ through a list;

$$\Phi = \text{\pcol@topfnotes} = (f_1, f_2, \dots, f_n) = \text{\vbox}\{f_1 f_2 \cdots f_n\}$$

where f_i is a `\vbox` containing the deferred footnote preceded by `\penalty\interlinepenalty` to allow \TeX to break footnotes to place them in two (or more) pages. That is, `\footnote` or `\footnotetext` in $p < p_t$ adds an element for the footnote to Φ , then all the elements¹¹⁶ are inserted to `\footins` by `\pcol@deferredfootins` invoked in `\pcol@restartcolumn` when it visits a column in p_t , or in `\pcol@startcolumn` when it starts a column-page in p_t . The macro `\pcol@output@end` also do the insertion by itself with merged footnote typesetting to let deferred footnotes be a part of post-environment stuff.

The reference to $\pi^f(p)$ for $p < p_t$ is also made in `\pcol@restartcolumn` and `\pcol@flushcolumn`. The former inserts $\pi^f(p)$ to `\footins` so that the column-page which the macro restarts is built as if it has the footnotes in $\pi^f(p)$ to make the column-page broken leaving the space for the footnotes. However, `\footins` is never grown because it has been fixed and thus additional footnotes will go to Φ as discussed above. Then `\footins` is discarded by `\pcol@makeecol` when the column-page completes, or by `\pcol@output@switch` when it leaves the column.

The reference to $\pi^f(p)$ by the latter macro `\pcol@flushcolumn` is to build the ship-out image of the column-page to be flushed. When this macro and other macros, namely `\pcol@makeecol` and `\pcol@makeflushedpage`, build the ship-out image in a page p having $\pi^f(p)$ using `\@makeecol`, we have to be careful of the fact that the column-page has been build as if it has footnotes in $\pi^f(p)$ but the footnotes are not included in its ship-out image but that of the page. Therefore, `\@colht` referred in `\@makeecol` should be shrunk by the sum of height and depth of $\pi^f(p)$ and `\skip \cdot \pi^f(p)` by `\pcol@shrinkcolbyfn`. Other and more subtle adjustment is to add the stretch and shrink factors of `\skip \cdot \pi^f(p)` at the tail of the column-page by `\pcol@unvbox@cc1v`. This is necessary because \TeX has broken the column-page taking account of the stretch and, more essentially, shrink factors, and thus without the factors the main vertical list in the column-page could be a little bit taller than `\@colht` causing overflow.

The feature gathering footnotes in all columns in a page brings a problem to explicit synchronization, because a column whose contents fit the top page at the last visit may be too tall on the synchronization because other columns have put some footnotes after the last visit. That is, we cannot simply build the top page combining $s_c(p_t)$ for all $c \in [0, C)$ and $\pi^f(p_t)$ because there could be $s_c(p_t)$ too tall to reside in p_t with $\pi^f(p_t)$.

To solve this problem, we perform the following operations prior to fix the contents of p_t having an explicit synchronization point in it. First one is *column-scan* to visit all columns by column-switching prior to the synchronization so that \TeX 's page builder has opportunities

¹¹⁵The column-page $s_c(p)$ could have some space at its bottom produced by, for example, `\newpage`, but exploitation of such space is extremely hard.

¹¹⁶More accurately, some trailing elements may be left in Φ if its total height is too large, as discussed in §11.5.

to break too tall column-pages. Since this scan could merely break footnotes rather than the main vertical lists in the column-pages and the broken footnotes will be reconnected when the `\output`-routine is invoked for the synchronization, we then examine if all $s_c(p_t)$ are accommodated in p_t with $\pi^f(p)$.

This examination for a synchronization by `\switchcolumn*` or its relatives is done as a part of the inherent synchronizing procedure to see if the combination of the tallest *top* items, i.e., top floats and the main vertical list, and the tallest *bottom* items, i.e., bottom floats and column-wise footnotes, is too large causing page flushing. As for page flushing and environment closing, this *pre-flushing column height check* requires a special kind of synchronized column-switching by which we flush pages up to $p_t - 1$ and examine if there is a too tall column.

Then if too tall columns are found, in either cases, we move to the *tallest* column to force a page break in the column so that we have a new page with shorter columns and shorter page-wise footnotes as well. In the synchronization by `\switchcolumn*` or its relatives, this forced page break is then applied to all other columns so that new column-pages have top floats, if any, below which we should place the synchronization point. This examination and forced page break is repeated until we have a page without any too tall columns, because a page break may bring deferred floats and footnotes which may result in a too tall column.

1.6 Text Coloring

1.6.1 Fundamental Mechanism

Text coloring done by `color` package and its relatives using `\special` stands on the fact that the main vertical list is *printed* in the order of occurrence in the source `.tex`. That is, a command such as `\color{red}` puts `\special{color push [1 0 0]}`¹¹⁷ into `.dvi` to make all stuff in the main vertical list colored red until other coloring `\special` inserted by other coloring macro appears in `.dvi`. This simple mechanism works well even when the pair of coloring `\specials` are in different pages and/or columns because, with respect to the main vertical list, everything between them in `.tex` is also surrounded by the `\special` pair in `.dvi`. As for other stuff such as header, footer, floats and footnotes, L^AT_EX surrounds them by `\color@begingroup` and `\color@endgroup` or other similar constructs so that they are colored without interference with the coloring of the main vertical list.

In `paracol` environment, however, the orders of the main vertical list in `.tex` and `.dvi` are not always same. When a column encounters a page break, in `.dvi` the other column should intervene between the stuff in the broken pair of column-pages possibly changing the color of the second column-page. A column-switching from c_1 to c_2 also makes the main vertical list out-of-order to cause another unexpected coloring because a coloring command in c_2 will have no effect when c_1 is revisited after that following its pre-switching stuff in `.dvi` which was put before the coloring. Therefore, we have to make *color contexts* in both `.tex` and `.dvi` coherent inserting appropriate `\specials` into `.dvi` whenever an out-of-order *jump* occurs in `.dvi` by a page break or in `.tex` by a column-switching.

The `color` package and its relatives¹¹⁸ assume that *printers* have a stack for coloring and thus a coloring `\special` pushes the new color into the stack while it is popped by another `\special` which will be inserted by `\aftergroup` mechanism when a group surrounding the coloring `\special` is closed. Therefore we have to keep track of the color context with *color stack*

$$\Gamma = (\gamma_1, \gamma_1, \dots, \gamma_n) = \text{\vbox}\{\gamma_1 \gamma_2 \dots \gamma_n\}$$

¹¹⁷If `.dvi` is processed by `dvips`, or other printer-dependent command corresponding to it.

¹¹⁸And all other coloring mechanism compliant with L^AT_EX 2_ε, hopefully and believingly.

where γ_i is a `\vbox` of 1 sp tall, 0 deep and 0 wide containing a coloring `\special` which `\set@color` puts into the main vertical list. That is, when `\set@color` is invoked we push γ to the tail of Γ , while when the corresponding `\reset@color` appears we pop it from Γ . Then when we encounter an out-of-order jump, at first we rewind the color stack in `.dvi` by putting `\specials` which `\reset@color` would put, and then reestablish the color stack by putting `\specials` in γ_i as if `\set@color` for it is invoked for all $\gamma_i \in \Gamma$. Therefore from the viewpoint of a *printer*, it will see stack-rewinding at the end of each column-page and the leaving points of column-switching, while the beginning of each column-page and the entry points of column-switching should have the sequence of coloring `\specials` to regain the color stack which the *printer* must have at each of the points.

In addition, for each column c we keep $\gamma_0^c = \text{\pcol@columncolor@box}\cdot c$ as the *default* color of the column c , optionally given by the API macro `\columncolor` or `\normalcolumncolor`. If given for c , it is assumed to be at the bottom of the color stack denoted by $\Gamma^c = (\gamma_0^c, \gamma_1, \dots, \gamma_n)$ which we rewind/reestablish at each out-of-order jump in the column c .

1.6.2 Coloring in Horizontal Mode

We have to pay attention to the fact a coloring command can appear in horizontal mode of course, and thus push/pop operations in a column-page would be done *before* the column-page starts when `\set@color` or `\reset@color` is in the second half of a page-crossing paragraph and if we immediately performed push/pop of the color stack in these macros. In addition, even in vertical mode these macros can appear before TeX finds a page break after which they must be in effect, if they are preceded by a sequence of non-breakable vertical items by which TeX's examination of the page break is *delayed* as well as the invocation of `\output` at the break.

In order to solve the problem of push/pop timing, we perform push/pop operations through `\insert` to our own register set `\pcol@colorins`. That is, we `\insert` γ to `\pcol@colorins` when we encounter a `\set@color` for γ , while its corresponding `\reset@color` also `\inserts` another `\vbox` γ^- of null-height/depth/width having a `\special` which the `\reset@color` puts into the main vertical list. Since we let `\count\pcol@colorins = 0` and `\skip\pcol@colorins = 0` to keep the `\insertion` from affecting the growth of `\pagetotal`, it is guaranteed that an inserted γ or γ^- is given to `\output` through `\pcol@colorins` together with `\box255` containing the corresponding `\special`.

When `\output` is invoked, `\pcol@colorins` has Γ_r containing γ_i and possibly its corresponding γ_i^- . Therefore, if `\output` is for a page break or a column-switching, we remove all pairs of γ_i and γ_i^- from `\pcol@colorins` to let it have Γ only with γ_j whose corresponding γ_j^- is not in Γ_r . For this removal, we scan Γ_r from its tail incrementing/decrementing a counter n_{pop} which we initialize to 0 before scanning. In the scan, we remove all γ^- unconditionally incrementing n_{pop} , and γ such that $n_{\text{pop}} > 0$ on the encounter with it decrementing n_{pop} . This scan is done by `\pcol@clearcolorstack`, invoked from `\pcol@opcol` for a page break and `\pcol@output@switch` for a column-switching through `\pcol@clearcst@unvbox`, and is for rewinding the color stack $(\gamma_0^c, \Gamma_r) = \Gamma_r^c$. Therefore, for each γ to be kept because of $n_{\text{pop}} = 0$ on the encounter with it we put `\special` for `\reset@color`. Note that on another scan for stack reestablishment, `\pcol@colorins` has Γ and is kept unchanged. Also note that other `\output` invocations such as that for floats do not touch Γ_r to allow it grows with γ and γ^- corresponding to `\set@color` and `\reset@color` in the column-page in which the invocation happens¹¹⁹.

The mechanism above especially for horizontal mode has subtle issues as follows.

¹¹⁹Unlike `\footins` which becomes void by putting its contents back to the main vertical list to reexamine the footnote placement possibly with splitting.

- If `\set@color` appears in a `\vbox`, the `\insertion` for pushing is not effective but corresponding `\reset@color` can be outside of the `\vbox` to make pushes and pops unbalanced because `\aftergroup` for it inserts it just after the closing of the `\vbox` if `\set@color` is not surrounded by an inner group.
- If we are in vertical mode, we can know if we are in a `\vbox` by `\ifinner`. However, in horizontal or math mode, `\ifinner` cannot help us because it is true iff we are in a `\hbox` or in an in-text math. In short, \TeX does not provide us with any convenient means to know if we are in a `\vbox`.

To solve the problem above, we introduced a trick with `\everyvbox` to turn a switch `\ifpcol@inner = true` at the beginning of every `\vbox` in a `paracol` environment, by which we suppress the `\insertion` for `\set@color` because a `\vbox` cannot cross a page boundary. As for that of `\reset@color`, we suppress it by not reserving our own macro `\pcol@reset@color@pop` for the `\insertion` by `\aftergroup`. That is, we reserve both `\reset@color` and `\pcol@reset@color@pop` with `\aftergroup` if we are outside of any `\vboxes`, while does the former only otherwise. By the same reason, we suppress the `\insertion` if we are in restricted horizontal mode, i.e., if both `\ifhmode` and `\ifinner` are true. On the other hand, we cannot suppress the `\insertion` when we are in an in-text math because it can cross a page boundary¹²⁰. Note that the detailed implementation shown in §12 does not interfere the use of `\everyvbox` inside/outside of `paracol` environments or is not affected by the use.

Another attention we should pay is that `\color` will leave `\aftergroup` tokens of `\reset@color` and thus they are invoked just after `\end{paracol}`. However, since we have completed all column-pages in the last page, the color stack in `.dvi` should be empty. Therefore to avoid stack underflow, we should reestablish Γ (not Γ^c) so that elements in the stack are popped by `\reset@color` invoked with the `\aftergroup` mechanism. We also take care of our own color stack popper `\pcol@reset@color@pop` which must do nothing, i.e., must not make an `\insertion`, after we completed the last page, i.e., if `\ifpcol@output` is *false*.

1.6.3 Changing Default Column Color

The implementation of `\columncolor` and `\normalcolumncolor` is relatively easy for the cases that they appear outside `paracol` environment or they define the default color of a column different from the current column. That is, for the default color of a column c we simply `\define \hat{\gamma}_0^c = pcol@columncolor \cdot c` to let it have what `\current@color` has for the color. Then, in `\begin{paracol}` in the former case or immediately in the latter, we let $\gamma_0^c = pcol@columncolor@box \cdot c$ have the coloring `\special` for the color acquiring an `\insert` from `\@freelist` if the box is \perp .

On the other hand, when the API commands are to define the default color of the current column c , we need to place the coloring at the bottom of color stacks in terms of `.tex` and `.dvi`. That is, for the former we have to rewind and reestablish the stack which can be different from Γ^c because the API command can follow a page break which \TeX does not yet find. Therefore, we maintain a *shadow* of Γ namely;

$$\hat{\Gamma} = \text{pcol@colorstack@shadow} = (\hat{\gamma}_1, \hat{\gamma}_2, \dots, \hat{\gamma}_n) = \backslash\@elt\{\hat{\gamma}_1\} \backslash\@elt\{\hat{\gamma}_2\} \cdots \backslash\@elt\{\hat{\gamma}_n\}$$

to which our version of `\set@color` pushes $\hat{\gamma}_i$ being `\current@color` which the original one defines, while popping is done automatically by \TeX 's grouping mechanism because pushes are

¹²⁰If an in-text math is in a `\hbox`, `\insertion` is not necessary because the math cannot cross a page boundary. Though we can detect it by a trick with `\everyhbox`, we abandon this idea because the request is not harmful. Another and more serious issue of coloring in math mode will be discussed shortly.

done by `\edef` rather than `\xdef`. Then before we `\define` $\hat{\gamma}_0^c$ we rewind $\hat{I}^c = (\hat{\gamma}_0^c, \hat{\gamma}_1, \dots, \hat{\gamma}_n)$ putting `\special` for pop to the main vertical list for each elements, and then after the `\definiton` of $\hat{\gamma}_0^c$ we reestablish \hat{I}^c putting coloring `\special` for each element.

As for placing γ_0^c at the bottom of Γ^c , we must ensure that the placement is done for the column-page in which the API command belongs to, as we did in ordinary push/pop of the color stack. Therefore the API command `\inserts` γ_0^c to Γ_r in the form of a `\vbox`, whose height and depth are `1sp` and width is 0, containing the coloring `\special` for γ_0^c . Then when Γ_r is scanned for rewinding in `\output`, this `\vbox` is found to let γ_0^c have the `\special` acquiring an `\insert` from `\@freelist` it was \perp . Note that Γ_r may have multiple `\vboxes` to update γ_0^c and if so the last one is effective.

1.6.4 Coloring in Math Mode

Unfortunately the solution above is imperfect because T_EX builds an implicit `\hbox` for a `{math stuff}` construct in math mode and an `\insert` in the construct does not contribute to the main vertical list at all¹²¹. Since the implicit `\hbox` does not care about `\everyhbox`, we cannot use the trick similar to that with `\everyvbox`. Another bad news is that built-in `\ifs` for mode checking cannot help us because we always have `\ifvmode = \ifhmode = false` and `\ifmmode = true` while `\ifinner` is `true` or `false` when we are in in-text or displayed math mode respectively. Therefore, we have to take care of the potential loss of `\insertion` for pushes and thus unmatched pops in Γ_r .

For example, we have to remember that, in the cases like `${\color{c}text}$` or `$\textcolor{c}{text}$` expanded to the former, the `\insertion` for push is lost while its counterpart for pop survives making it necessary to check the existence of pushing counterpart for each pop in Γ_r ¹²². Note that the fact that the pop in the examples is in the in-text math does not help us, because the pop in `$$\begingroup\color{c}text\endgroup$` is also in the in-text math while its pushing counterpart performs an effective `\insertion`, and two `\insertions` must be presented in Γ_r because we can have a page-break in `text`. Therefore, we have to find a means to examine whether a pop γ_i^- has its counterpart γ_i in Γ_r to remove γ_i from Γ_r if exists or to ignore γ_i^- otherwise. That is, we have to attach an identifier m to γ_i and γ_i^- , i.e., to make them $\gamma_{i,m}$ and $\gamma_{i,m}^-$.

Since the only means we have for the communication with `\output` routine is what we `\insert` to Γ_r , the `\inserted` `\vbox` must carry an identifier m for a push/pop in math mode. To do that, we make `\vbox` m sp wide ($m > 0$) if our version of `\set@color` is in math mode to represent $\gamma_{i,m}$ and $\gamma_{i,m}^-$, while the width is 0 otherwise as described in §1.6.2. Then in the scan of Γ_r for rewinding in `\output`, we suppress incrementing/decrementing n_{pop} for $\gamma_{i,m}$ and $\gamma_{i,m}^-$, but remove $\gamma_{i,m}$ if $\gamma_{i,m}^-$ is in Γ_r as a successor while we keep it in Γ_r otherwise putting a `\special` of pop for it to the main vertical list.

To ensure that $\gamma_{i,m}$ has its counterpart $\gamma_{i,m}^-$ in Γ_r iff the push and pop are in a column-page, we maintain the counter `\pcol@mcid` incremented before (the attempt of) the `\insertion` of $\gamma_{i,m}$ with $m = \text{\pcol@mcid}$ and the `\aftergroup` reservation for that of $\gamma_{i,m}^-$. Then the counter is zero-cleared by `\output` routine in order to keep it less than `\pcol@mcpushlimit = 1000` unless, roughly speaking, a column-page has a unexpectedly large number of math constructs having coloring commands in them. Note that this zero-clearing does not ensure that an identifier m is unique in Γ_r . That is, it can happen that Γ_r has $\gamma_{i,m}$, $\gamma_{i,m}^-$, $\gamma_{j,m}$ and/or $\gamma_{j,m}^-$ in this order for $i < j$, when two math constructs with coloring for i and j are in different

¹²¹The contents is not thrown away but `\insertion` itself is added to the list rather than given to `\output`.

¹²²Since a pop is always in a group one level outer from its push counterpart, the pop request should be presented if the push does.

paragraphs and `\output` is invoked at or after the end of the paragraph with the math for i . This potential duplication is, however, unharmed because of the following.

- Since a math construct cannot have immediate `\output` invocations in it, the order of the elements in Γ_r must be $\gamma_{i,m}$, $\gamma_{i,m}^-$, $\gamma_{j,m}$ and $\gamma_{j,m}^-$ from its bottom to top, though some of them could be missing. Therefore, if $\gamma_{i,m}^-$ is in Γ_r , then $\gamma_{j,m}$ must follow it if it exists not causing accidental matching with $\gamma_{i,m}^-$.
- If $\gamma_{i,m}$ is in Γ_r but $\gamma_{i,m}^-$ is not, it means we have a page break between vertical items corresponding to $\gamma_{i,m}$ and $\gamma_{i,m}^-$ to keep the `\insertion` of $\gamma_{i,m}^-$ and anything following it from appended into Γ_r . Therefore, Γ_r cannot have $\gamma_{j,m}^-$ not causing accidental matching with $\gamma_{i,m}$.

1.6.5 Emptiness of a Column-Page

The mechanism above works well with respect to coloring, but it has a problem that a column-page created by, for example, a forced page break may not be perfectly empty but can have some coloring `\specials` for color stack reestablishing and rewinding. They are of course invisible but affect the examination of column-page emptiness for explicit synchronization. That is, we examine if a column-page does not have anything by a tricky way by `\pcol@ifempty` but the existence of coloring `\specials` makes the examination failed even if no other ordinary stuff such as boxes and skips are in the column-page.

Therefore we need to put coloring `\specials` for color stack establishing and rewinding a little bit more carefully to avoid empty column-pages just having such `\specials` as follows. When we start a new column-page, we don't put `\specials` for establishing immediately but save the color stack Γ^c into $\Gamma_s = \text{\pcol@colorstack@saved}$. Then when we leave the column-page by switching or page breaking, we examine the emptiness of the column-page and if so we do nothing, while otherwise we put the `\specials` for reestablishing Γ_s at the top of the column-page and those for rewinding Γ_r at the bottom. Similarly, when we revisit a column-page, we examine its emptiness and if so we save Γ^c into Γ_s , while otherwise we put `\specials` for reestablishing Γ^c and nullify Γ_s so that nothing will be put at the top of the column-page when we leave it. By these mechanisms, an empty column-page should not have coloring `\specials`, while non-empty ones should have a sequence of triples; reestablishing `\specials`; ordinary main vertical list items including coloring `\specials` inserted by `\color` etc.; and then rewinding `\specials`.

1.7 Parallel-Paging, Column-Swapping, Column-Separating Rule Drawing and Background Painting

We have the following four extensions, which are correlated to each other, from the basic parallel-columning.

Parallel-paging to extend the concept of parallel-columning in a page to a pair of adjacent pages. A *left* parallel-page starts from column-0, has C_L columns where C_L is given by the first optional argument of `\begin{paracol}`, while a *right* parallel-page starts from column- C_L and has $C - C_L$ columns. Since we let $C_L = C$ when parallel-paging is not in effect, we may ship out columns $c \in [0, C_L)$ always and then, if $C_L < C$, ship out columns $c \in [C_L, C)$ as a right parallel-page.

The pair of parallel-pages can be *paired* to comprise a virtual page p and thus has common page number $page(p)$, while *non-paired* parallel-paging produces two individual pages from a

internal page p (i.e., set of all columns $\{c \mid c \in [0, C)\}$) whose left and right components have page numbers $page(p)$ and $page(p) + 1$ respectively. Since a page p is internally considered as the set of all columns $c \in [0, C)$ always, regardless of paired or non-paired parallel-paging, the difference between them arises only in two-sided ship-out process in which the header, footer and left-margin are common for left/right paired parallel-pags while they have to depend on the parity of the number of each non-paired parallel-page. Note that `paracol` does not specify the parity of a left non-paired parallel-page number, but the number is decided by the page from which a parallel-paged `paracol` environment starts.

In ship-out process, we build the ship-out image of a right parallel-page in our own `\box` register `\pcol@rightpage` instead of the usual `\@outputbox`. The register, however, must *survive* after `\end{paracol}` to keep the columns in the last right parallel-page, so that it is shipped out when the whole of last page including post-environment stuff is shipped out, or, more complicatedly, to be passed to the next `paracol` environment as a part of its pre-environment stuff.

Page-wise stuff spanning all columns, i.e., spanning stuff being pre-environment stuff or page-wise floats, spanning texts, page-wise footnotes and post-environment stuff are always placed in a left parallel-page, while the corresponding regions for them in a right parallel-page are always blank¹²³ unless pre-environment stuff has the last page of the previous `paracol` environment.

Column-swapping to reverse the order of columns in even numbered pages from left-to-right to right-to-left. It is enabled by the specifier ‘c’ of `\twosided`¹²⁴. Though it is fundamentally simple because we just need to reverse the scanning order of columns from left-to-right (i.e., 0 to $C - 1$) to right-to-left (i.e., $C - 1$ to 0) in the ship-out process of an even numbered page, there are a few complications in the implementation of related functionalities.

First, a paired parallel-page should also be swapped so that a *physical* left (resp. right) parallel-page has columns $C - 1$ to C_L (resp. $C_L - 1$ to 0) in this order. Note that this parallel-page swapping also swaps the page in which page-wise stuff are placed. That is, if both paired parallel-paging and column-swapping are in effect, page-wise stuff are placed in the physical right parallel-page, or in other words they always placed in the page in which column-0 resides. Note that since column-swapping with non-paired parallel-paging is meaningless and thus column-swapping is disabled.

Second, the side margin to which a marginal note goes can be swapped but enabling this swap is independent of column-swapping and done by the specifier ‘m’ of `\twosided`, though almost all users will specify both swapping consistently. Since the side margin for a marginal note is decided in `\output` routine by `\pcol@addmarginpar` being our own version of L^AT_EX’s macro for marginal notes `\@addmarginpar`, the page in which the marginal note resides has been fixed. However, the number of the page and thus its parity may not have been fixed yet due to the possible jump in column-0 taking place afterward, unlike column-swapping for which the page number has been fixed because it is performed in ship-out process. Since it is too costly to avoid this possibly wrong placement, we have to accept the possibility as L^AT_EX itself does. Also unlike column-swapping, the swapping of marginal notes is not disabled in non-paired parallel-paging because it is meaningful.

Another remark for marginal notes is that two ore more columns may *share* a margin, inevitably if a (parallel) page has three or more columns or intentionally with a setting of `\marginparthreshold`. Therefore, the context of marginal notes cannot be in column-context

¹²³So far. In some future, we could implement a special setting to let pre-environment stuff, post-environment stuff and page-wise footnotes are split into both parallel-pages, and to make it possible that a page-wise float or a spanning text has its counterpart placed in the corresponding right parallel-page.

¹²⁴Or the backward compatible macro `\swapcolumninevenpages`.

but should be in page context, or cannot simply give the bottom of the last marginal note (i.e., L^AT_EX's `\@mparbottom`) but should show all marginal notes in margins in a page¹²⁵. Therefore, each page context has all marginal notes in the form of lists of their top and bottom positions in all margins as $\pi^m(p)$, so that we find a space for a marginal note in a column to add it to not only to the bottom but also into a space between two marginal notes having already been put by other columns.

Third and finally, we have to take care the placement of spanning texts. In version 1.2 to which column-swapping is introduced, we let a spanning text belong to column- $(C - 1)$ instead usual column-0 so that its left edge is aligned to the left edge of the leftmost column, i.e., that of the text area. However this simple solution has a severe problem that, if a spanning text is broken into two pages, its second half should be put in the rightmost column. In addition, even when a spanning text does not have page break in it, such wrong placement may happen if the text is followed by `\nobreak` and thus a page break is made above the text but *after* the text is processed.

In version 1.3, this problem is solved by capturing the first half of a spanning text in `\output` routine for the page break in the text, and the second half or the whole of it in that for synchronized column-switching to close the text. Since an invocation of `\output` routine means that it has been fixed which page the spanning text or its part resides in, we can place the text much more reliably expecting the parity of the page number has also been fixed. In addition, this decision making in `\output` routine allows (or forces) us to let spanning texts always belong to column-0 preserving the consistency of, for example, local counter values referred to in them, while we need to shift a text to the left edge of the text area if it resides in an even numbered page. Furthermore, this spanning text capturing enables to measure the vertical size of the captured text together with the vertical position of its top edge to record them in the list $\pi^s(p)$, so that we draw column-separating rules skipping the text and painting its background with a specific color different from colors of columns and column-separating gaps, as discussed shortly.

Column-separating rule drawing to draw a vertical rule in *column-separating gap* is correlated with a part of *background painting* to paint each region in a page with a color specific to the region. Thanks to the list of spanning texts $\pi^s(p) = (\text{span}(H_i, h_i))_i^n$, we can draw column-separating rules skipping spanning texts in the page p as the sequence of;

$$\begin{aligned} & \text{rule}(H'_1), \text{gap}(h_1), \dots, \text{rule}(H'_n), \text{gap}(H'_n), \text{rule}(H'_{n+1}) \\ & H'_i = H_i - (H_{i-1} + h_{i-1}) \quad H_0 = h_0 = 0 \quad H_{n+1} = \pi^h(p) \end{aligned}$$

where $\text{rule}(H')$ is a vertical rule of H' high and $\text{gap}(h)$ is a vertical space of h . A rule may be colored with the color specified by `\colseprulecolor` for each column-separating gap or all of them. Note that if column-swapping is in effect, a column c is *preceded* by c -th column-separating gap which may have its own width and color for its rule, rather than being followed by it.

Background painting also uses the list $\pi^s(p)$ to paint the background of each column- c with the color B_c^c , each column-separating gap following the column- c with B_g^c , and spanning texts with B_s and B_S , where $B_a^{[c]}$ is specified by the second argument of `\backgroundcolor{a[c]}{color}` ($a \in \{c, g, s, S\}$) and kept in the macro `\pcol@bg@color@a[@c]`. The region to be painted for each item is as follows where $[(x_0, y_0)(x_1, y_1)]$ means the region $\{(x, y) \mid x \in$

¹²⁵Before version 1.3, we have `\@mparbottom` in column-context because a column has its own area for marginal notes, which can be the gap between columns rather than a margin of a page.

$[x_0, x_1), y \in [y_0, y_1)$ of the top-down xy -coordinate whose origin is at the left-top corner of the leftmost column.

$$\begin{aligned}
R_c^c(i) &= [(W_c, H_{i-1} + h_{i-1}) (W_c + w_c, H_i + d_{c/g})] \\
R_g^c(i) &= [(W_c + w_c, H_{i-1} + h_{i-1}) (W_{c+1}, H_i + d_{c/g})] \\
R_{\{s,S\}}(i) &= [(0, H_i) (W_T, H_i + h_i + d_s)] \\
W_c &= \sum_{d=c_0}^{c-1} (w_c + g_c) \quad c_0 = \begin{cases} 0 & c < C_L \\ C_L & c \geq C_L \end{cases} \quad W_T = \text{\texttt{\textbackslashtextwidth}} \\
d_{c/g} &= \begin{cases} \text{\texttt{\textbackslashmaxdepth}} & i = n + 1 \wedge H'_{n+1} > 0 \wedge \text{non-last page} \\ 0 & \text{otherwise} \end{cases} \\
d_s &= \begin{cases} H_{n+1} - (H_n + h_n) + \text{\texttt{\textbackslashmaxdepth}} & i = n \wedge H'_{n+1} = 0 \wedge \text{non-last page} \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

In the specifications above, w_c and g_c is the width of the column c and that of the column-separating gap following it, defined by `\columnratio` or `\setcolumnwidth` and stored in `\pcol@columnwidth·c` and `\pcol@columnsep·c` respectively. The additions of $d_{c/g}$ and d_s are to extend the bottom edge of each region down to the bottom of text area. In addition, for each $R_a^{[c]} = [(x_0, y_0)(x_1, y_1)]$, *extensions* $e_a^{[c]}(\{x, y\}^\pm)$ can be specified to shift the base points x_0, y_0, x_1 and y_1 left (x^-), right (x^+), upward (y^-) and downward (y^+) respectively. That is, a region is defined as;

$$R_a^{[c]} = [(x_0 - e_a^{[c]}(x^-), y_0 - e_a^{[c]}(y^-)) (x_1 + e_a^{[c]}(x^+), y_1 + e_a^{[c]}(y^+))]$$

with the optional shifts specified by the first argument of `\backgroundcolor` as $\{a[c](x^\pm, y^\pm)\}$ (for both x^-/y^- and x^+/y^+) or $\{a[c](x^-, y^-)(x^+, y^+)\}$ and kept in macros `\pcol@bg@ext@d·@·a[·@·c]` where $d \in \{1, r, t, b\}$ for x^- (1), x^+ (r), y^- (t) and y^+ (b). Note that $e_a^{[c]}(\{x, y\}^\pm)$ can be extremely large, namely greater than or equal to 9000pt, to mean the region is extended to a border near by the corresponding paper edge. More specifically, by this *infinite extension*, each xy coordinate in $[(x_0, y_0)(x_1, y_1)]$ is defined as follows to represent a coordinate being 10000 pt $- e_a^{[c]}(\{x, y\}^\pm) + \text{\texttt{\textbackslashpagemargin}}$ inside from the page edge;

$$\begin{aligned}
x_0 &= -W_M + (10000 \text{ pt} - e_a^{[c]}(x^-) + W_R) \\
y_0 &= -(H_S + H_M) + (10000 \text{ pt} - e_a^{[c]}(y^-) + H_R) \\
x_1 &= (W_P - W_M) - (10000 \text{ pt} - e_a^{[c]}(x^+) + W_R) \\
y_1 &= (H_P - H_S - H_M) - (10000 \text{ pt} - e_a^{[c]}(y^+) + H_R) \\
W_P &= \text{\texttt{\textbackslashpaperwidth}} \\
W_M &= \text{\texttt{\textbackslashlin}} + \begin{cases} \text{\texttt{\textbackslashoddsidemargin}} & \neg f_{\text{even}} \\ \text{\texttt{\textbackslashevensidemargin}} & f_{\text{even}} \end{cases} \\
W_R &= H_R = \text{\texttt{\textbackslashpagemargin}} \\
H_P &= \text{\texttt{\textbackslashpaperheight}} \\
H_M &= \text{\texttt{\textbackslashheadsep}} + \text{\texttt{\textbackslashheadheight}} + \text{\texttt{\textbackslashtopmargin}} + \text{\texttt{\textbackslashlin}} \\
H_S &= \text{height}(\pi^b(p)) + \text{depth}(\pi^b(p))
\end{aligned}$$

where f_{even} is *true* iff we are in an even numbered page and two-sided typesetting is specified by the optional argument of `\documentclass` or by the specifier ‘p’ of `\twosided` explicitly or implicitly.

Another remark is that column-swapping affects $R_c^c(i)$ and $R_s^c(i)$ to *mirror* the region making a reflection-symmetric transformation on it using a vertical edge of a page as the axis. That is, $[(x_0, y_0)(x_1, y_1)]$ for a region is transformed to $[(W_T - x_1, y_0)(W_T - x_0, y_1)]$ if $x_{\{0,1\}}$ is not extended infinitely. With infinite extension on the other hand, before this transformation x_0 and/or x_1 are calculated by the rule above replacing W_M with $W_P - (W_M + W_T)$ to represent the width of the right margin rather than the left.

On the other hand, the mirroring of $R_{\{s,S\}}$ is enabled by the specifier ‘b’ of `\twosided`, together with other regions being top margin (t, T), bottom margin (b, B), left margin (l, L), right margin (r, R), page-wise floats (f, F) and page-wise footnotes (n, N). The geometrical specifications R_a for those regions are given as follows, but the coordinate origin is at the top-left corner of text area (rather than the leftmost column).

$$\begin{aligned}
R_{\{t,T\}} &= [(-W_M + W_R, -H_M + H_R), (W_P - W_M - W_R, 0)] \\
R_{\{b,B\}} &= [(-W_M + W_R, H_T - H_M), (W_P - W_M - W_R, H_P - H_M - H_R)] \\
R_{\{l,L\}} &= [(-W_M + W_R, 0), (0, H_T)] \\
R_{\{r,R\}} &= [(W_T, 0), (W_P - W_M - W_R, H_T)] \\
R_{\{f,F\}} &= [(0, 0), (W_T, H_S)] \\
R_{\{n,N\}} &= [(0, H_T - H_N), (W_T, H_T)] \\
H_T &= \text{\code\texttheight} + \text{\code\maxdepth} \\
H_N &= \text{\code\skip\footins} + \text{\code\ht\footins} + \text{\code\dp\footins}
\end{aligned}$$

Note that, since we use text area coordinates, in the calculation of infinite extension H_S is let be 0.

We have other regions for columns and column-separating gaps, namely R_C^c and R_G^c , which vertically span all over text area regardless existence of any page-wise stuff. Therefore, their geometrical specifications are as follows with text area coordinates.

$$\begin{aligned}
R_C^c &= [(W_{c-1}, 0), (W_{c-1} + w_c, H_T)] \\
R_G^c &= [(W_{c-1} + w_c, 0), (W_c, H_T)]
\end{aligned}$$

In addition, we have to paint pre-environment stuff and post-environment stuff with color $B_{\{p,P\}}$. The region $R_{\{p,P\}}$ for them is defined as follows with text area coordinates where H_B is the y -coordinate of the bottom of previous `paracol` environment if any, or 0 otherwise.

$$R_{\{p,P\}} = \begin{cases} [(0, H_B), (W_T, H_S)] & \text{pre-environment stuff} \\ [(0, H_B), (W_T, H_T)] & \text{post-environment stuff} \end{cases}$$

Note that painting of post-environment stuff is done *outside* `paracol` environment when the post-environment stuff encounters a page break, unless another `paracol` environment starts in the page and thus the post-environment stuff becomes pre-environment stuff of the second (or subsequent) environment.

Finally, we define the order of background painting as follows, where a , $a(i)$, a^c and $a^c(i)$ mean R_a , $R_a(i)$, R_a^c and $R_a^c(i)$ respectively, so that a succeeding region is *overlaid* on preceding regions.

$$\begin{aligned}
&T \rightarrow B \rightarrow L \rightarrow R \\
&\rightarrow G^0 \rightarrow \dots \rightarrow G^{C-2} \rightarrow C^0 \rightarrow \dots \rightarrow C^{C-1} \\
&\rightarrow t \rightarrow b \rightarrow l \rightarrow r \rightarrow N \rightarrow n \rightarrow \{F, P\} \rightarrow \{f, p\}^{126} \\
&\rightarrow S(1) \rightarrow \dots \rightarrow S(n)
\end{aligned}$$

$$\begin{aligned}
&\rightarrow g^0(1) \rightarrow \cdots \rightarrow g^{C-2}(1) \rightarrow c^0(1) \rightarrow \cdots \rightarrow c^{C-1}(1) \rightarrow s(1) \\
&\rightarrow \cdots \\
&\rightarrow g^0(n) \rightarrow \cdots \rightarrow g^{C-2}(n) \rightarrow c^0(n) \rightarrow \cdots \rightarrow c^{C-1}(n) \rightarrow s(n) \\
&\rightarrow g^0(n+1) \rightarrow \cdots \rightarrow g^{C-2}(n+1) \rightarrow c^0(n+1) \rightarrow \cdots \rightarrow c^{C-1}(n+1)
\end{aligned}$$

1.8 Page-wise Float Placement

In the release on 2015/01/10, L^AT_EX's float placement mechanism was drastically changed to avoid *out-of-order* appearance of page-wise floats as follows. Since the cause of overtaking of a page-wise float by a column-wise float is that they are in two separated lists `\@dbldeferlist` for the former and `\@deferlist` for the latter, in the new implementation the two lists are merged to let all floats go to `\@deferlist`. To distinguish page-wise and column-wise floats in the list, `\end@dblfloat` lets the page-wise float processed by the macro have a special depth of `1sp`, while depth of column-wise floats are 0 since `\@endfloatbox` add a `\vskip` of 0 at the end of the box of floats.

Then all float placement macros invoked in `\output-routine` examine the depth of floats in the list they are working on against a newly introduced macro `\f@depth` by also newly introduced `\@testwrongwidth`, so that they process only floats of a page/column category specified by `\f@depth`, while those not matching to `\f@depth` are let go to `\@deferlist` to inhibit succeeding floats of the same type from overtaking them. The `\definition` of `\f@depth` is done only by modified `\@dblfloatplacement`, always invoked in a group, to let it have `1sp` so that float placement macros usually work on column-wise ones with the default setting of `\f@depth = \z@` except for special occasions in which the placement of page-wise floats is tried.

Though the mechanism should work well with *ordinary* multi-columned documents, it is incompatible with `paracol` almost inherently. That is, in the first place we have to separate float-related lists into the sets of them corresponding to columns as we do¹²⁷. Therefore, it is obviously nonsense to merge the list for page-wise floats, i.e., `\@dbldeferlist`, to `\@deferlist` of a particular column, and thus we have to stick with the conventional implementation to process page-wise and column-wise floats separately as follows.

- (1) We `\define` our own `\end@dblfloat` namely `\pcol@end@dblfloat` whose definition is exactly same as the old version of `\end@dblfloat`, and replace the new version with it by `\letting` them equivalent in `\pcol@zparacol` by which start `paracol` environments. Therefore, page-wise floats composed in a `paracol` environment is processed in the traditional way, i.e., being included in `\@dbldeferlist` rather than `\@deferlist` and having ordinary depth 0.
- (2) Each invocation of `\@dblfloatplacement` in our own `\output-routine` is followed by a `\let` to nullify the setting of `\f@depth = 1sp` done by `\@dblfloatplacement` by doing `\f@depth=\z@`. By this setting, `\@tryfcolumn` in `\pcol@startpage` and `\@makefcolumn` in `\pcol@output@clear` work on their argument `\@dbldeferlist` in the way exactly same as in 2014 or before.

¹²⁶In column flushing, the order is $\{F, P\} \rightarrow \{f, p\} \rightarrow N \rightarrow n$ but this reversion should have no effect (almost always).

¹²⁷If counters `figure` and `table` are global and we have to avoid inter-column overtaking with respect to, for example, the partial ordering rooted by the top-left corner, merging column-wise lists together with the merge of `\@deferlist` and `\@dbldeferlist` might be a solution to let the depth of a column-wise float be $c\text{sp}$ while that of page-wise is $C\text{sp}$. However such implementation is not only tough but also doubtful to be worthwhile.

- (3) Among L^AT_EX’s macros in its `\output`-routine which we use in our own one as well, only `\@addtodblcol` changed its target from `\@dbldeferlist` to `\@deferlist`. That is, if the macro fails to put a page-wise float to the page we just have started by `\pcol@startpage`, the float is added to `\@deferlist` rather than `\@dbldeferlist`. Therefore, when we apply `\@sdblcolelt` to the copy of `\@dbldeferlist` to invoke `\@addtodblcol` for each of its element float, we have to save `\@deferlist` somewhere, to `\reserved@c` in reality, and clear it prior to the application. Then after all elements are processed, we have to let `\@dbldeferlist` have what `\@deferlist` have, while `\@deferlist` should regain its original contents from the saved place. A subtle issue is that we might work with L^AT_EX of 2014 or older in which the floats are returned to `\@dbldeferlist`. Therefore to make `paracol` compatible with both of new and old versions, we have to *add* `\@deferlist` to `\@dbldeferlist` rather than replacing `\@dbldeferlist` with `\@deferlist`. This addition should work well, because we clear both lists before the application of `\@sdblcolelt` and then one of them will have the still-deferred floats after the application while the other remains empty.
- (4) We convert `\@deferlist` to `\@dbldeferlist` in `\pcol@output@start` to start a `paracol` environment, and perform the reverse operation in `\pcol@output@end` to close the environment. Though it is very unlikely (or maybe impossible) that the `\@deferlist` imported in the former operation has L^AT_EX’s (i.e., not `paracol`’s) double-column floats of 1sp deep, we make such floats old-fashioned making their depth 0 so that they can be put in a page built in the `paracol` environment. On the other hand, the latter cannot *export* a list having floats of 1sp deep because they have been produced in the closing `paracol` environment or have passed our *custom* `\pcol@output@start` when they were imported¹²⁸.

Note that the operations (1), (2) and (4) are fully compatible with 2014 or older version of L^AT_EX, because with the old version; (1) `\pcol@end@dblfloat` is equivalent to `\end@dblfloat`; (2) modification of `\f@depth` cannot be seen because it does not exist; and (4) we virtually do nothing in the importation. As for (3), we explicitly take care of the compatibility as shown above.

2 Interaction with T_EX and L^AT_EX

The macros of `paracol` interacts with T_EX and L^AT_EX through various registers and macros as discussed in this section.

2.1 Registers

2.1.1 Insertion Registers

`\footins` is used to `\insert` footnotes through it by `\footnote` and `\footnotetext`, and then in `\output` routine the footnotes `\inserted` in a page is presented in the register. The register is referred to by the following macros.

- `\pcol@makecol` examines if the register has page-wise footnotes and, if so, saves it into $\pi^f(p)$ if $p = p_t$ or discards it otherwise.

¹²⁸Therefore, if one try to *smuggle* a double-column float of the new scheme into a `paracol` and to pass it through the environment to another double-column world, the float will become a single-column one. Even if such guy a really exists and complains about this transformation, however, we have good right to say “don’t do that”.

- `\pcol@startcolumn` inserts $\pi^f(p)$ into the column-page to be created through the register.
- `\pcol@specialoutput` logs the contents of the register for debugging.
- `\pcol@output@start` examines if the register has footnotes to be merged with those in `paracol` environment, refers to its height plus depth to calculate effective `\@colht`, and/or inserts its contents through itself to the main vertical list as the first part of merged footnotes.
- `\pcol@makenormalcol` combines footnotes in the register to other pre-environment stuff to make a spanning stuff, or makes save/restore of the register to/from `\@tempboxa` to exclude footnotes from spanning stuff when merged-footnote type-setting is specified.
- `\pcol@output@switch` saves the register into $\pi^f(p)$ or $\kappa_c(\tau)$, or discards its contents, when we leave from the column c with footnotes.
- `\pcol@restartcolumn` restores $\kappa_d(\tau)$ or $\pi^f(p)$ to the register and then inserts the contents of `\box\footins` into itself so that it contributes to the main vertical list to be rebuilt for the column d .
- `\pcol@getcurrfoot` for column d lets the register have $\kappa_d(\tau)$.
- `\pcol@savefootins` saves the register into an `\insert` for $\pi^f(p)$ or $\kappa_c(\tau)$.
- `\pcol@deferredfootins` refers the `\skip` component of the register to have the vertical skip above page-wise footnotes and inserts deferred footnotes through the register.
- `\pcol@output@end` inserts $\pi^f(p)$ into the last page through the register.
- `\pcol@fntexttop{text}` inserts the footnote $\langle text \rangle$ and a penalty through the register.
- `\pcol@fntextbody{text}` refers to the `\skip` component of the register to cap the height of the footnote $\langle text \rangle$.

`\bx@A, \dots, \bx@R` have floats created by `\xfloat` in the ordinary usage of *fundamental* L^AT_EX of 2014 or earlier or that without the extension of e-T_EX or its relatives. On the other hand, in L^AT_EX of 2015 or later and with e-T_EX or its relatives, the set is `\bx@A, \dots, \bx@Z, \bx@AA, \dots, \bx@ZZ`. In addition to the use in L^AT_EX, we use these registers for completed column-pages $s_c(p)$ (`\pcol@opcol`, `\pcol@flushcolumn`), main vertical list $\kappa_c(\beta)$ (`\pcol@output@start`, `\pcol@output@switch`, `\pcol@flushcolumn`) and column-wise footnotes $\kappa_c(\tau)$ (`\pcol@output@switch`) in current column-pages, spanning stuff including pre-environment stuff $\pi^i(p)$ (`\pcol@startpage`, `\pcol@output@start`) and page-wise footnotes $\pi^f(p)$ (`\pcol@makecol`, `\pcol@output@switch`) in pages, MVL-floats for main vertical lists in synchronized pages (`\pcol@synccolumn`), and page-wise floats deferred from `paracol` to post-environment stuff (`\pcol@output@end`).

2.1.2 Integer Registers

`\deadcycles` is T_EX's primitive register to count the number of `\output` requests made between two `\shipout` operations. It is zero-cleared by `\pcol@invokeoutput` because it can have a large number in a `paracol` environment.

`\outputpenalty` is T_EX's primitive register to have the page-break penalty with which `\output` routine is invoked. It is referred to by `\pcol@output` to know whether it has special code

less than -10000 , and by `\pcol@specialoutput` in detail for the dispatch according to the code. The register is also used for the communication from the latter, which lets it be -10000 for our own special `\output` routines, to the former to determine `\vsize` according to if the register has a value greater than -10004 or not.

`\interlinepenalty` is T_EX's primitive register to have the page-break penalty inserted between two lines. The register is referred to in the following macros.

- `\pcol@output@start` to make pre-environment merged footnotes followed by this `\penalty` on the `\insertion`, and to insert it to start the first column-page allowing page-break before the start unless it is inhibited by `\if@nobreak = true`.
- `\pcol@restartcolumn` to insert this `\penalty` to resume a column-page allowing page-break if `\if@nobreak = false`.
- `\pcol@deferredfootins` to let the second half of split Φ have this `\penalty` as the very first element.
- `\pcol@fntexttop{text}` to make the footnote $\langle text \rangle$ followed by this `\penalty` on the `\insertion`.
- `\pcol@fntextother{text}` to make the footnote $\langle text \rangle$ preceded by this `\penalty` in Φ .
- `\pcol@fntextbody` to let the register have `\interfootnotelinepenalty`.

`\floatingpenalty` is T_EX's primitive register to have the page-break penalty added to `\insertpenalties` if an `\insert` is moved to the page next to the page in which the line having the `\insert` resides. It is let have 20000 in `\pcol@fntextbody` for footnote typesetting.

`\vbadness` is T_EX's primitive register to have the threshold of the badness of `\vbox` construction with underfull messages. That is, if the badness exceeds the threshold on a `\vbox` construction, T_EX will complain showing an underfull message. In `\pcol@makenormalcol` and `\pcol@deferredfootins`, the register is temporarily let have 10000 to avoid that `\@makecol` invoked in the former and `\vsplit` done in the latter causes the message with inevitable underfull.

`\showboxdepth` is T_EX's primitive register to determine the maximum depth of box structure to be shown in logging etc. The register is let have 10000 in `\pcol@ShowBox` for full logging.

`\showboxbreadth` is T_EX's primitive register to determine the maximum breadth of box structure to be shown in logging etc. The register is let have 10000 in `\pcol@ShowBox` for full logging.

`\interfootnotelinepenalty` is an API `\count` register to have `\interlinepenalty` for footnotes. It is used in `\pcol@fntextbody` to let `\interlinepenalty` have it.

`\@one` is a `\chardef` register to have 1 . The register is referred to by the following macros mainly for incrementing another register.

```
\pcol@F@count, \pcol@output, \pcol@opcol, \pcol@setpnoelt,
\pcol@nextpage, \pcol@nextpelt, \pcol@startpage, \pcol@checkshipped,
\pcol@outputelt, \pcol@ioutputelt, \@outputpage, \pcol@bg@paint@ii,
\pcol@output@start, \pcol@makenormalcol, \pcol@output@switch,
```

`\pcol@setcurrcol`, `\pcol@iscancst`, `\pcol@addmarginpar`,
`\pcol@do@mpbout@i`, `\pcol@sync`, `\pcol@flushcolumn`,
`\pcol@measurecolumn`, `\pcol@synccolumn`, `\pcol@makeflushedpage`,
`\pcol@imakeflushedpage`, `\pcol@iflushfloats`, `\pcol@freshpage`,
`\pcol@output@end`, `\pcol@invokeoutput`, `\pcol@zparacol`,
`\pcol@setcolwidth@r`, `\pcol@setcolwidth@s`, `\pcol@setcw@scan`,
`\pcol@setcw@calcf`, `\pcol@synccounter`, `\pcol@com@syncallcounters`,
`\pcol@stepcounter`, `\pcol@com@switchcolumn`, `\pcol@sptext`,
`\pcol@visitallcols`, `\pcol@ifootnote`, `\pcol@ifootnotemark`.
`\pcol@swapcolumn`, `\pcol@set@color@push`,

`\tw@` is a `\chardef` register to have 2. It is used in `\pcol@setcurrcol` to let $\kappa_c(\sigma) = 2$ when `\if@nobreak = true` but `\if@afterindent = false`, in `\pcol@setcw@calcf⟨x⟩⟨y⟩⟨z⟩` to calculate $x \cdot 2^k$, $y/2^k$ and $(x/y) \cdot 2^k$ with various k , and in `\pcol@swapcolumn` to calculate $C^1 - (c' - C^0) - 2 = c - 1 = c^g$ for the column-separating gap ordinal c^g physically following the column c at the position c' .

`\m@ne` is a `\count` register to have -1 . It is used in the following macros.

- `\pcol@setpnoelt`, `\pcol@nextpelt`, `\pcol@getpelt` and `\pcol@setmpbelt` to decrement `\@tempcnta` which initially has $p - p_b$ for a page p .
- `\pcol@bg@paint@i` to decrement C_b^1 by one locally to have the column scanning range $[C_b^0, C_b^1 - 1]$.
- `\pcol@iscancst` to decrement n_{pop} by one.
- `\pcol@do@mpbout@i` to let `\@tempcnta` have it to indicate left margin.
- `\pcol@setcolwidth@r` to calculate $C^1 - C^0 - 1$.
- `\pcol@setcw@calcf⟨x⟩⟨y⟩⟨z⟩` in `\@whilenum` loops to calculate $z'/2^k$ and $z'/2k - 16$ where $z'/2^k \approx x/y$.
- `\pcol@iadjustfnctr` to decrement `\c@footnote`.

`\sxt@@n` is a `\chardef` register to have 16. It is used in `\pcol@setcw@calcf⟨x⟩⟨y⟩⟨z⟩` to calculate $Z = z \times 1 \text{pt} = z' \cdot 2^{16-k}$ where $z'/2^k \approx x/y$.

`\@m` is a `\mathchardef` register to have 1000. It is used in `\pcol@synccolumn` and `\pcol@output@end` to let `\prevdepth = 1000pt` on a synchronization or the closing `paracol` environment with an empty main vertical list, and in `\pcol@setcw@getspec@i` to add 1000pt to stretch and shrink components of `\@tempskipa` having a column/gap specification to make it sure the skip register has those components.

`\@M` is a `\mathchardef` register to have 10000. It is used in the following macros

- `\pcol@ShowBox` to let `\showboxdepth` and `\showboxbreadth` have 10000 for full logging of a box.
- `\pcol@output` to examine if `\outputpenalty < -10000` to mean a special `\output` request.
- `\pcol@specialoutput` to let `\outputpenalty = -10000` to tell `\pcol@output` that the special `\output` request is our own.
- `\pcol@makenormalcol` and `\pcol@deferredfootins` to let `\vbadness` have 10000 temporarily to avoid underfull messages.

- `\pcol@synccolumn` to bias `\pcol@textfloatsep` by `10000pt` to indicate a column-page has an MVL-float and in `\pcol@cflt` and `\pcol@addflhd` to remove the bias.
- `\pcol@switchcol` and `\pcol@flushclear` to put `\penalty-10000` for forced page break.
- `\pcol@setcw@calcf⟨x⟩⟨y⟩⟨z⟩` to let $Z = z \times 1pt = 10000pt$ if x/y is too large.

`\@Mii` is a `\mathchardef` register to have 10002. It is used in `\pcol@end@dblfloat` to examine if `\@floatpenalty = -10002` to mean the float environment to be closed is given in horizontal mode.

`\@Miv` is a `\mathchardef` register to have 10004. It is used in `\pcol@output` to examine if `\outputpenalty = -10004` for a dummy `\output` request made by L^AT_EX's float-related macros and our `\pcol@invokeoutput` to ensure the real request is not eliminated when it is made at the very beginning of a page or a column-page. It is also used in `\pcol@specialoutput` for footnote logging when `\outputpenalty = -10004`.

`\@MM` is a `\mathchardef` register to have 20000. It is used in `\pcol@fntextbody` to let `\floatingpenalty` have it for footnote typesetting.

`\@beginparpenalty` is a `\count` register to have the page-break penalty inserted before the first `\item` of each list-like environment. The penalty is determined in class files and is, for example, `-\@lowpenalty = -51` with `article.cls`. It is referred to and inserted by `\pcol@zparacol` when it finds the `paracol` environment to start is at the very beginning of a list-like environment.

`\@floatpenalty` is a `\count` register to have the penalty code -10002 or -10003 given by `\@xfloat` at the beginning of a float environment according to the environment is in horizontal or vertical mode respectively, or by `\marginpar` for a marginal note in the same meaning. It is referred to by `\pcol@end@dblfloat` to insert the penalty, and by `\pcol@xympar` to confirm `\marginpar` is error free.

`\@topnum` is a `\count` register to have the maximum number of top floats which the current column-page can accept further. It is used in `\pcol@setcurrcol` and `\pcol@iigetcurrcol` to save/restore it into/from $\kappa_c(\nu_t)$. The macro `\pcol@synccolumn` also lets `\@topnum = 0` to inhibit top-float insertions in the current column-page any more after a synchronization.

`\@botnum` is a `\count` register to have the maximum number of bottom floats which the current column-page can accept further. It is used in `\pcol@setcurrcol` and `\pcol@iigetcurrcol` to save/restore it into/from $\kappa_c(\nu_b)$.

`\@colnum` is a `\count` register to have the maximum total number of floats which the current column-page can accept further. It is used in `\pcol@setcurrcol` and `\pcol@iigetcurrcol` to save/restore it into/from $\kappa_c(\nu_c)$.

`\col@number` is a `\count` register to have the number of columns. It is let have 1 by `\pcol@zparacol` and `\pcol@sptext` regardless the real number of columns C in order to keep `\maketitle` from putting the title by `\twocolumn`.

`\c@page` is L^AT_EX's counter `page` being a `\count` register to have the page number. It is referred to by `\pcol@setpnoelt`, and `\pcol@output@start` to let $\pi^p(p) = page(p)$. The macro `\pcol@startpage` reload the register from $\pi^p(p_t-1)$ and then increment it by one

usually but two in non-paired parallel-paging, and repeat $\pi^p(p_t) = page(p_t)$ and incrementing $page(p)$ for each float pages of page-wise floats. Reloading $page(p)$ to the register from $\pi^p(p)$ is also done by `\pcol@getpelt` for macros using `\pcol@getcurrpage`, and by `\pcol@outputelt`, `\pcol@sync` and `\pcol@makeflushedpage` by `\pcol@getcurrpinfo`. Then the register is referred to by the following macros to examine its parity.

- Our own `\@outputpage` to give $page(p)$ or $page(p) + 1$ to `\pcol@outputpage@l` and `\pcol@outputpage@r` which finally let the register have the value to be referred to by `\pcol@@outputpage` being L^AT_EX's `\@outputpage`.
- `\pcol@bg@swappage` to determine the values of `\pcol@bg@leftmargin` and `\ifpcol@bg@@swap` with other factors.
- `\pcol@shiftspanning` to decide the necessity of shifting spanning text left with column-swapping, examining `raw \c@page` at the `\output` request to close the spanning text rather than $\pi^p(p_t)$ which will have the correct value with respect to possible jump *after* the macro completes its work.
- `\pcol@addmarginpar` to determine the margin to which a marginal note goes.
- `\pcol@do@mpbout@i` to determine which of M_L^l or M_L^r is the target of the operation specified by `\pcol@do@mpbout@elem`.
- `\pcol@swapcolumn⟨c1⟩⟨c2⟩⟨C0⟩⟨C1⟩` to determine c_2 for c_1 if column-swapping is in effect.

In addition, to do the parity examination in `\pcol@bg@swappage` above correctly, the macros `\pcol@ioutputelt`, `\pcol@makeflushedpage`, `\pcol@imakeflushedpage`, `\pcol@iflushfloats` and `\pcol@output@end` temporarily increment the register by one when they are working on a right non-paired parallel-page.

The other users are `\localcounter{ctr}` to check $\langle ctr \rangle \neq page$, `\pcol@remctrelt` to let $\c1@theta = \pcol@stepcounter{\theta}$ unless $\theta = page$, and `\pcol@FF` to write it to a log file as a part the logging information of memory leak debugging.

`\c@footnote` is L^AT_EX's counter `footnote` being a `\count` register to have the footnote number. It is referred to by `\pcol@zparacol` to let $b_f = \pcol@footnotebase$ have its value, by `\pcol@calcfncptr` to calculate the footnote ordinal which the one of its invoker `\pcol@iadjustfncptr` sets into the register, and by `\endparacol` to let the register have $b_f + n_f = \pcol@footnotebase + \pcol@nfootnotes$.

`\c@theta` is a `\count` register being L^AT_EX's counter θ . It is referred to by `\pcol@savectrelt` to let $val_c(\theta) = val(\theta)$ for saving it in Θ_c , by `\pcol@cmpctrelt` to examine if $val_0(\theta) = val(\theta)$ to detect a modification outside `paracol` environment, and by `\pcol@syncctrelt` to let $val_c(\theta) = val(\theta)$ for all c for counter synchronization. It is also referred to by `\pcol@remctrelt` and `\localcounter` to examine if $\theta = page$. The macros `\pcol@setctrelt` and `\pcol@stpdlzelt` restore the value of the counter from $val_c(\theta)$, while `\pcol@stpclelt` lets $val(\theta) = 0$ for zero-clearing of descendent counters.

`\count@` is a `\count` register for temporary use. It is used in `\pcol@iscancst` to have m of $\gamma_{i,m}$, `\pcol@addmarginpar` to have the physical column position of the current column c in which `\marginpar` is given, and in `\pcol@extract@fil@i⟨s⟩\@nil` to extract the unit of the stretch component s of a glue.

`\@tempcnta` is a `\count` register for temporary use. The usages of this register are as follows.

- In `\pcol@F@count`, it is used to measure the cardinality of `\@freelist`.

- In `\pcol@makecol`, it is used to keep $page(p_t)$ gotten by `\pcol@getcurrpinfo` until we store it back by `\pcol@defcurrpage`.
- In `\pcol@setpageno`, `\pcol@setpnoelt`, `\pcol@nextpage`, `\pcol@nextpelt`, `\pcol@getcurrpage`, `\pcol@getpelt`, `\pcol@addmarginpar` and `\pcol@setmpbelt`, it has $p - q$ when we scan $\pi(q)$ for all $q \in [p_b, p_t]$ and the current column-page is at p .
- In `\pcol@checkshipped`, it has c when we scan S_c for all $c \in [0, C]$ to examine if all of them are not empty and thus we have pages to be shipped out.
- In `\pcol@ioutputelt`, it has c' when we scan $s_c(q)$ for all $c \in [0, C_L)$ or $c \in [C_L, C]$ to build the shipping out image of a page q , where $c = c'$ or $c = C^1 - 1 - (c' - C^0)$ where $(C^0, C^1) \in \{(0, C_L), (C_L, C)\}$.
- In `\@outputpage`, it has $page(p)$ or $page(p) + 1$ to be given to `\pcol@outputpage@l` or `\pcol@outputpage@r` as their argument when they are used to ship out the second (not always right) component of a parallel-page pair.
- In `\pcol@bg@columnleft`, it has a value in $[C_b^0, c)$ to sum up the width of columns and column-separating gaps in the range.
- In `\pcol@output@switch`, it is used to have $page(p)$ obtained by `\pcol@getcurrpinfo` and simply to store the value in $\pi^p(p)$ by `\pcol@defcurrpage` when we use these macros to add an element to $\pi^s(p)$ and/or let $\pi^f(p) = \text{\footins}$.
- In `\pcol@setcurrcol`, it has the code calculated from `\if@nobreak` and `\if@afterindent` to be saved in $\kappa_c(\sigma)$.
- In `\pcol@scancst<box>`, it is let have $\langle box \rangle \in \{\text{\pcol@colorins}, \text{\pcol@colorstack@saved}\}$ and then is referred to in `\pcol@iscancst`.
- In `\pcol@addmarginpar` besides the page scan shown above, it is used to scan all columns whose physical position is left from the current column c to measure the distance between the left edges of the leftmost column and c .
- In `\pcol@do@mpbout@i`, it has ± 1 according to the margin (left = -1) which marginal notes outside `paracol` environments goes to.
- In `\pcol@flushcolumn`, it is used to throw $page(p_t)$ away when we get it by `\pcol@getcurrpinfo` because we don't need it.
- In `\pcol@setcolwidth@r`, it has c to scan fractions r_d where $d = c - C^0$ in its argument $\langle ratio \rangle$ specified by `\columnratio`, and then to distribute the unspecified portion of `\textwidth` evenly to w_c for all $c \in [C^0 + k + 1, C^1)$, where k is the number of fractions and $(C^0, C^1) \in \{(0, C_L), (C_L, C)\}$.
- In `\pcol@setcw@scan(C^0)\langle C^1 \rangle\{spec\}`, it has c for two loops for $c \in [C^0, C^1)$ to add `' , '` to the tail of $\langle spec \rangle$ as many as $k = C^1 - C^0$ and then to process first k elements in $\langle spec \rangle$, and is referred to by `\pcol@setcw@set` invoked in the second loop.
- In `\pcol@setcw@calcf\langle x \rangle\langle y \rangle\langle z \rangle`, it used to calculate and to have k such that $z'/2^k \approx x/y$.
- In `\pcol@cmpctrelt`, it has $val(\theta)$ of a counter θ to be compared with $val_0(\theta)$.
- In `\pcol@com@switchcolumn`, it has $(c + 1) \bmod C$ being the target of column-switching.
- In `\pcol@sptext`, it temporarily has d being the target of column-switching during we let `\pcol@nextcol` have 0 to visit the leftmost column to put a spanning text.

- In `\pcol@visitallcols`, it has $d \in [0, C) - \{c\}$ being the columns to be visited for column-scanning.
- In `\definecolumnpreamble{c}{pream}`, c is assigned to the register to ensure c is a number.
- In `\pcol@calcfncctr`, it has the footnote ordinal calculated by the macro to be referred to by the invokers `\pcol@iadjustfncctr` and `\pcol@iifootnotetext`.
- In `\marginparthreshold{t_l}[t_r]` it is let have t_l , while in the related macro `\pcol@marginparthreshold[t_r]` it is let have t_r .

`\@tempcntb` is a `\count` register for temporary use. It is used in the following macros.

- In `\pcol@ioutputelt` it has c , while in `\pcol@imakeflushedpage` and `\pcol@iflushfloats` it has c' , to have $c = c'$ or $c = C^1 - 1 - (c' - C^0)$ according to column-swapping for c' -th iteration of column scanning loop for $c' \in [C^0, C^1)$, where $(C^0, C^1) \in \{(0, C_L), (C_L, C)\}$.
- In `\pcol@scancst` and `\pcol@iscancst`, it has n_{pop} .
- In `\pcol@addmarginpar`, it is let have the column number d whose physical position is left from the current column c to measure the distance between the left edges of the leftmost column and c .
- In `\pcol@sync` and `\pcol@measurecolumn`, it has the (so-far) tallest column which gives V_P .
- In `\pcol@setcolwidth@r`, it has $C^1 - C^0 - 1$ then $C^1 - 1$ and finally $C^1 - \min(C^0 + k, C^1 - 1)$, where k is the number of fractions given in the argument of `\columnratio` and $(C^0, C^1) \in \{(0, C_L), (C_L, C)\}$.
- In `\pcol@setcw@calcf{x}{y}{z}`, at first it is used to calculate $z'/2^k \approx x/y$ and then to calculate $Z = z \times 1 \text{ pt} = z' \cdot 2^{16-k}$.
- In `\pcol@visitallcols`, it has $c = \text{\pcol@currcol}$ to exclude it from the column-scan targets.

2.1.3 Dimension Registers

`\vsize` is \TeX 's primitive register to have the height of a page or a column-page being built. It is let be `\@colroom` or `\maxdimen` by `\pcol@output`.

`\hsize` is \TeX 's primitive register to have the width of a page or a column-page being built. It is let be w_c by `\pcol@invokeoutput` to restart (or stay in) the column c , be `\textwidth` by `\pcol@sptext` for spanning text, and be either of `\textwidth` or w_c by `\pcol@fntextbody` according to the footnote typesetting being page-wise or column-wise respectively.

`\maxdepth` is \TeX 's primitive register to have the maximum depth of the page being built. In \LaTeX , it is assumed that its value is fixed at `\begin{document}`, in which the value is saved into `\@maxdepth`, for the typesetting throughout the document, unless a bottom float is added to a page in which the register is let have 0 until it is shipped out. This temporary setting for a page with bottom floats has some reasonability but its implementation for `paracol` environments having column-switching from/to a column-page with bottom floats to/from another one without them is too costly¹²⁹. Therefore, we

¹²⁹That is, we would need to incorporate `\maxdepth` as a member of column-context, but we don't because the idea of temporary setting itself is too vague to pay the effort and a precious membership in column-context.

let the register have `\@maxdepth` in `\pcol@output` and `\pcol@combinefloats` to cancel the temporary setting done in `\@addtobot` for the references by T_EX's page builder and `\@cflt` respectively.

`\boxmaxdepth` is T_EX's primitive register to have the maximum depth of `\vboxes`. The macros `\pcol@cflt`, `\pcol@opcol`, `\pcol@ioutputelt`, `\pcol@combinefloats`, `\pcol@output@flush` and `\pcol@output@clear` let it be `\@maxdepth` for the boxes having a completed column-page or page to cap the depth of the box. The macro `\pcol@makecol⟨d⟩` also do that when $d = \@maxdepth$ but it can be $d = 0$ when it is invoked to build last page.

`\splitmaxdepth` is T_EX's primitive register to have the maximum depth of the `\vbox` being the first half of a box being split. It is used in `\pcol@deferredfootins` to cap the depth of the first half of deferred footnotes split from Φ , and in `\pcol@fntextbody` to let it have the depth of `\strutbox`.

`\prevdepth` is T_EX's primitive register to have the depth of the box which just has been added to a vertical list, or to be given to T_EX's vertical list builder for the calculation of the vertical skip inserted below the last box. The macro `\pcol@invokeoutput` refers to it to save its value in `\pcol@prevdepth` before putting a dummy `\vbox` and making a `\output` request, and then let it have `\pcol@prevdepth` which is given by `\output` routine for the column that we resume. The macro `\pcol@zparacol` also refers to it to save it in `\pcol@firstprevdepth` for the pre-environment stuff.

`\vfuzz` is T_EX's primitive register to have the threshold height of overfull messaging. It is set to 0 in `\pcol@ShowBox⟨b⟩` to ensure overfull for any `\box` b of non-null height.

`\maxdimen` is a `\dimen` register to have 16383.99999pt being the largest legal dimensional value. The usages of this register are as follows.

- For the `\insert` register set `\pcol@colorins = Lr`, `\dimen\pcol@colorins` is let be `\maxdimen` for the consistency with our intention that a column-page can have virtually infinite number of `\insertions` for text coloring.
- In `\pcol@output`, it is set into `\vsize` when `\outputpenalty = -10004` for the dummy `\output` request so that no page break should occur between the dummy and real requests.
- Our own `\dimen` register `\pcol@textfloatsep` has `\maxdimen` if a column-page does not have synchronization points to let top floats are inserted in usual way. Therefore, `\pcol@floatplacement` and `\pcol@zparacol` let the register have `\maxdimen` as the initial value. Then the macros `\pcol@makecol` and `\pcol@combinefloats` examine if `\pcol@textfloatsep = \maxdimen` to determine the operation type of top float insertion, while `\pcol@synccolumn` does that to know if it is flushing a column-page with a synchronization point or is setting the first synchronization point in the column-page. The macro `\pcol@addflhd` also examines it for the measurement of the combined height of top and bottom floats, while `\pcol@measurecolumn` gives it `\maxdimen` as its third argument for bottom floats.
- The page context $\pi^h(p)$ has $-\maxdimen$ if the page is a float page. The macro `\pcol@startpage` makes that when it creates such a page.
- Our own `\pcol@prevdepth` and that saved in $\kappa_c(\delta)$ have `\maxdimen` if the main vertical list is empty at a synchronization. The macro `\pcol@measurecolumn` makes that when it finds an empty list.

- The column-context $\kappa_c(\rho_t)$ may have `\maxdimen` if the column c has a float column in the last page of a `paracol` environment and the floats in it can be put as top floats. The macro `\pcol@makefcolumn` makes this special assignment, and `\pcol@flushcolumn` and `\pcol@imakeflushedpage` examine it.
- The macro `\pcol@makefolelt` let the room for floats in a float page be $-\maxdimen$ if it finds no further floats can be added to the page.
- At a synchronization, we measure the maximum combined size of top floats and the main vertical list V_T , that of footnotes and bottom floats V_B , that of four items V_P , and V'_P being V_P or $V_P + \text{textfloatsep}$ according to the existence of bottom floats. We also let D_T and D_P be the minimum depth of the column-pages which gives V_T and V'_P respectively. For the measurement, the macro `\pcol@sync` lets $V_T = V_B = V_P = V'_P = -\maxdimen$ and $D_T = D_P = \maxdimen$ as their initial values. Then the macro `\pcol@synccolumn` examines if $D_T = \maxdimen$ to mean the synchronization point is set just below the top floats of a column whose main vertical list is empty. On the other hand, `\pcol@makeflushedpage` and `\pcol@output@end` examine if $V'_P = -\maxdimen$ to mean the last column-pages are empty.

`\linewidth` is a `\dimen` register for an API parameter of \LaTeX to have the width of a line possibly shorter than `\columnwidth` in `list`-like environments. It is let have $w_c - \mu$ by `\pcol@invokeoutput` for the outermost paragraphs in `paracol` environment, where $\mu = \text{pcol@lrmargin} = \text{textwidth} - l$ which `\pcol@zparacol` lets have to represent the left-plus-right margin of the `list`-like environment, whose `\linewidth` is l , enclosing `paracol` if any. The macro `\pcol@sptext` also sets the register temporarily for spanning texts but letting it have $\text{textwidth} - \mu$.

`\footnotesep` is a `\dimen` register for an API parameter of \LaTeX to have the vertical space inserted in a footnote when it is split into two or more pages. It is used in `\pcol@fntextbody` to `\splittopskip` have it, and to make the first line of the footnote is at least as tall as the amount in the register.

`\topmargin` is a `\dimen` register for an API parameter of \LaTeX to have the width (height) of the top margin minus 1 inch. The register is used as an element of `\pcol@bg@pagetop` to calculate the distance from the origin at the left-top corner of text area to the top edge of a page. The other users are `\pcol@ioutputelt` and `\pcol@makeflushedpage` temporarily add the height-plus-depth of $\pi^b(p)$ to the register to make the calculation biased shifting the origin to the left-top corner of column area, i.e., below $\pi^b(p)$. The macro `@outputpage` also refers to the register together with `\headheight` and `\headsep` to calculate the distance from the page top (ignoring 1 inch shift) to the text area top.

`\oddsidemargin` is a `\dimen` register for an API parameter of \LaTeX to have the width of the left margin minus 1 inch for two-sided odd-numbered pages and all single-sided pages. The register is used together with `\evensidemargin` in `\pcol@outputpage@l`, `\pcol@outputpage@r` and `\pcol@bg@swappage` to decide the left margin of the page they are working on.

`\evensidemargin` is a `\dimen` register for an API parameter of \LaTeX to have the width of the left margin minus 1 inch for two-sided even-numbered pages. The register is used together with `\oddsidemargin` in `\pcol@outputpage@l`, `\pcol@outputpage@r` and `\pcol@bg@swappage` to decide the left margin of the page they are working on.

`\headheight` is a `\dimen` register for an API parameter of \LaTeX to have the height of page headers. The register is used together with `\topmargin` and `\headsep` as an element

of `\pcol@bg@pagetop` and in `\@outputpage` to calculate the distance from the real and imaginary page top to the text area top respectively.

`\headsep` is a `\dimen` register for an API parameter of \LaTeX to have the vertical distance from the bottom of a page header to the text area top. The register is used together with `\topmargin` and `\headheight` as an element of `\pcol@bg@pagetop` and in `\@outputpage` to calculate the distance from the real and imaginary page top to the text area top respectively.

`\textheight` is a `\dimen` register for an API parameter of \LaTeX to have the height of text area in a page. The register is referred to by `\pcol@output` to examine if a page is very short, by `\pcol@getpinfo`, `\pcol@startpage`, `\pcol@flushfloats` and `\pcol@output@end` to let `\@colht` be it for a page without spanning stuff (so far), by `\pcol@outputelt`, `\@outputpage`, `\pcol@output@flush` and `\pcol@output@clear` to build a page to be shipped out, by `\pcol@bg@textheight` to calculate H_T , and by `\pcol@fntextbody` to cap the height of a footnote.

`\textwidth` is a `\dimen` register for an API parameter of \LaTeX to have the width of a page, which we occasionally refer to as W_T . The register is referred to by `\pcol@ioutputelt`, `\pcol@imakeflushedpage` and `\pcol@iflushfloats` to build a `\hbox` of `\textwidth` wide to have all columns (in a left or right parallel-page). It also referred to by following macros; `\pcol@bg@swappage` to calculate the right margin for mirrored background painting; `\pcol@bg@r`, `\pcol@bg@f`, `\pcol@bg@n`, `\pcol@bg@p` and `\pcol@bg@s` to specify the width of background of page-wise stuff to be painted; `\pcol@shiftspanning` to calculate the left-shift amount of a spanning text in column-swapping; `\pcol@addmarginpar` to measure the distance between the right edges of the rightmost and current columns; `\pcol@zparacol` for the calculation of $\mu = \pcol@lrmargin$; `\pcol@setcolwidth@r` for the calculation of w_c for all $c \in [0, C_L)$ or $c \in [C_L, C)$; `\pcol@setcw@calc factors` for the calculation of W_T/W and $(W_T - W)/F$; `\pcol@sptext`, `\footnoterule` of `paracol`'s local and `\endparacol` to set it in `\columnwidth`; and `\pcol@fntextbody` to set it in `\hsize` if page-wise footnote typesetting is in effect.

`\columnwidth` is a `\dimen` register for an API parameter of \LaTeX to have the width of a column. The register is let have w_c by `\pcol@getcurrcol` for the column c , then is referred to by the following macros; `\pcol@shiftspanning` to calculate the left-shift amount of a spanning text in column-swapping; `\pcol@addmarginpar` to measure the distance between the right edges of the rightmost and current columns; `\pcol@imakeflushedpage` and `\pcol@iflushfloats` to put each column-page into a `\hbox` of w_c wide for shipping a page out; `\pcol@invokeoutput` to let `\linewidth` and `\hsize` have the value of or based on it; and `\pcol@fntextbody` to let `\hsize` have it for column-wise footnote typesetting. The register is also let have `\textwidth` by `\footnoterule` of `paracol`'s local defined in `\pcol@zparacol` if page-wise footnote typesetting is in effect, by `\pcol@sptext` for spanning texts, and by `\endparacol` for post-environment stuff.

`\columnsep` is a `\dimen` register for an API parameter of \LaTeX to have the width of column-separating gaps. It is referred to by `\pcol@setcolwidth@r` to calculate w_c for all $c \in [0, C_L)$ or $c \in [C_L, C)$, and by `\pcol@setcw@getspec` as the default width of column-separating gaps.

`\columnseprule` is a `\dimen` register for an API parameter of \LaTeX to have the width of the rules to be drawn in column-separating gaps. It is referred to by `\pcol@buildcolseprule` and `\pcol@buildcselt` to draw the rule, and by `\pcol@hfil` to examine if it is positive

to mean the rule is really drawn and if so to add skips of $-\backslash\text{columnseprule}/2$ to surround the rule to nullify the width of the rule.

`\marginparwidth` is a `\dimen` register for an API parameter of \LaTeX to have the width of a marginal note. It is temporarily modified by `\pcol@addmarginpar` so that a left marginal note is placed appropriately.

`\marginparsep` is a `\dimen` register for an API parameter of \LaTeX to have the distance between a marginal note and text area. It is temporarily modified by `\pcol@addmarginpar` so that a right marginal note is placed appropriately.

`\marginparpush` is a `\dimen` register for an API parameter of \LaTeX to have the minimum vertical distance between two marginal notes. It is referred to by `\pcol@addmarginpar` to find a place for a marginal note and remember the place in $\pi^m(p)$.

`\paperheight` is a `\dimen` register for an API parameter of \LaTeX to have the height of physical pages H_P . It is referred to by `\pcol@bg@paperheight` to calculate $H_P - 2W_R$.

`\paperwidth` is a `\dimen` register for an API parameter of \LaTeX to have the width of physical pages W_P . It is referred to by `\pcol@bg@swappage` to calculate the right margin for mirrored background painting, and by `\pcol@bg@paperwidth` to calculate $W_P - 2W_R$.

`\z@` is a `\dimen` register to have 0pt to initialize `\pagerim`, `\belowfootnoteskip` and `\skip\pcol@colorins` at their declarations, and is used in the following macros.

```
\pcol@ShowBox, \pcol@makecol, \pcol@combinefloats, \pcol@nextpelt,
\pcol@floatplacement, \pcol@startpage, \pcol@restartcolumn,
\pcol@outputelt, \pcol@ioutputelt, \pcol@buildcolseprule,
\pcol@buildcset@S, \pcol@buildcset, \@outputpage,
\pcol@startcolumn, \pcol@bg@paint@i, \pcol@bg@paintregion,
\pcol@output@start, \pcol@putbackmvl, \pcol@iscancst,
\pcol@deferredfootins, \pcol@combinefootins, \pcol@addmarginpar,
\pcol@getmparbottom, \pcol@sync, \pcol@measurecolumn,
\pcol@synccolumn, \pcol@makeflushedpage, \pcol@imakeflushedpage,
\pcol@output@end, \pcol@invokeoutput, \pcol@setcolwidth@s,
\pcol@setcw@calc@factors, \pcol@setcw@calc@f, \pcol@extract@fil@ii,
\pcol@sptext, \pcol@fntextbody. \pcol@marginpar, \pcol@icolumncolor,
\pcol@set@color@push, \pcol@reset@color@pop,
\pcol@reset@color@mpop, \pcol@backgroundcolor@iii.
```

It is also used to give the number 0 for the initializations of `\pcol@currcol`, `\pcol@ncol`, `\pcol@ncolleft` and `\count\pcol@colorins` at their declarations, and in the following macros.

```
\pcol@ShowBox, \pcol@F@count, \pcol@makecol, \pcol@opcol,
\pcol@setpnoelt, \pcol@nextpelt, \pcol@checkshipped, \pcol@getpelt,
\pcol@outputelt, \pcol@ioutputelt, \@outputpage, \pcol@startcolumn,
\pcol@output@start, \pcol@output@switch, \pcol@setcurrcol,
\pcol@iscancst, \pcol@addmarginpar, \pcol@setmpbelt,
\pcol@do@mpbout@i, \pcol@sync, \pcol@synccolumn,
\pcol@makeflushedpage, \pcol@imakeflushedpage, \pcol@flushfloats,
\pcol@iflushfloats, \pcol@freshpage, \pcol@output@end,
\pcol@zparacol, \pcol@setcolwidth@r, \pcol@setcw@calc@f,
```

`\pcol@synccounter`, `\pcol@com@syncallcounters`, `\pcol@stepcounter`,
`\pcol@stpclelt`, `\pcol@com@switchcolumn`, `\pcol@switchcolumn`,
`\pcol@sptext`, `\pcol@switchcol`, `\pcol@visitallcols`, `\pcol@xympar`,
`\endparacol`.

`\p@` is a `\dimen` register to have 1pt. It is used in `\pcol@ShowBox`, `\pcol@cflt`, `\pcol@addflhd`, `\pcol@synccolumn`, `\pcol@output@end`, `\pcol@setcolwidth@s`, `\pcol@setcw@getspec@i`, `\pcol@setcw@fill` and `\pcol@setcw@calcf`, and the top level assignment to `\@tempkipa` for the invocation of `\pcol@defkw`, as the shorthand of pt.

`\@totalleftmargin` is a `\dimen` register to have the total size of the left margins of a list-like environment and those surrounding it. It is given to `\parshape` by `\pcol@invokeoutput` and `\pcol@sptext` if `paracol` is enclosed in a list-like environment.

`\@themargin` is a control sequence `\let`-equal to `\evensidemargin` for two-sided even numbered pages or `\oddsidemargin` for others. In `\pcol@outputpage@l` and `\pcol@outputpage@r` it is bound to one of `\dimen` registers for the references in `\pcol@outputpage@ev`¹³⁰.

`\@maxdepth` is a `\dimen` register to have `\maxdepth` at `\begin{document}` to recover the temporary update of `\maxdepth` with 0 by `\@addtobot` for bottom float incorporation in a page. As discussed in the explanation of `\maxdepth`, in `paracol` environments `\maxdepth` is let have `\@maxdepth` always by the assignments in `\pcol@output` and `\pcol@combinefloats`. Other users, `\pcol@cflt`, `\pcol@opcol`, `\pcol@ioutputelt`, `\pcol@combinefootins`, `\pcol@output@flush` and `\pcol@output@clear`, let `\boxmaxdepth` have `\@maxdepth` so as to limit the depth of boxes for a completed column-page or page to the value for page typesetting, while `\pcol@flushcolumn` and `\pcol@imakeflushedpage` do that by `\pcol@@makecol` giving the register to it. The other usage of the register is to calculate background painting parameter H_T by `\pcol@bg@textheight`, and to determine the bottom edge of the backgrounds of columns and column-separating gaps through the argument of `\pcol@buildcolseprule` given by `\pcol@ioutputelt`, `\pcol@imakeflushedpage` and `\pcol@iflushfloats`. The register is also referred to by `\pcol@unvbox@cclv` to go back the last baseline of the main vertical list in `\box255`, and by `\pcol@deferredfootins` to let `\splitmaxdepth` have its value to cap the depth of the first half of footnotes split from Φ .

`\@colht` is a `\dimen` register to have the height of columns in a page possibly shrunk from `\textheight` by spanning stuff. The usages of the register are as follows.

- In `\pcol@startpage`, `\pcol@output@start`, `\pcol@flushfloats` and `\pcol@output@end`, it is initialized to `\textheight`. In first two, the value of the register is reduced to reflect spanning stuff if exists and then set into $\pi^h(p)$, while the setting by the third is referred to by its callee `\pcol@iflushfloats`.
- In `\pcol@getpelt`, `\pcol@sync`, `\pcol@flushcolumn`, `\pcol@makeflushedpage` and `\pcol@imakeflushedpage`, it is let have $\pi^h(p)$. In addition `\pcol@sync` examines if $\@colht < V_T + V_B + v(f)$, and `\pcol@makefcolum` uses it to initialize the room of a float column as well as the height of $\kappa_c(\beta)$ for it.
- In `\pcol@opcol`, it is used to add `\pcol@clearcolorstack` to the bottom of $\kappa_c(\beta)$ whose height is `\@colht`.

¹³⁰The reference in `\pcol@@outputpage` being L^AT_EX's `\@outputpage` is done after the macro itself makes the assignment, which is of course consistent with those in our macros.

- In `\pcol@startcolumn(*)`, `\pcol@flushcolumn(*)` and `\pcol@freshpage`, it is used to let `\@colroom` have it.
- In `\pcol@restartcolumn(*)`, it is saved and restored for the use as the height cap of deferred footnote `\insertion` in `\pcol@deferredfootins` because it can be shrunk by the non-deferred page-wise footnotes.
- In `\pcol@output@flush` and `\pcol@output@clear`, it is given to `\pcol@makeflushedpage` as its argument. The macro `\pcol@makeflushedpage(*)` lets `\@colht` be the argument if it is less than `\@colht` and thus is given by `\pcol@output@end`.

In addition, in the macros with ‘(*)’ above and `\pcol@makecol`, the register is passed to `\pcol@shrinkcolbyfn` to shrink the height in it temporarily to keep the space required to put page-wise footnotes in the page they are working on, for the reference by starred macros themselves or `\@makecol` invoked in `\pcol@makecol`.

`\@colroom` is a `\dimen` register to have the height of a column possibly shrunk from `\@colht` by top and bottom floats. The register is initialized to have `\@colht` by `\pcol@startcolumn`, `\pcol@output@start`, `\pcol@flushcolumn` and `\pcol@freshpage`, the last three of which also save it into $\kappa_c(\beta^r)$. This save operation is also done by `\pcol@output@switch` while restoring from it done by `\pcol@restartcolumn`, but the latter macro may shrink the amount in its callee `\pcol@putbackmvl` to capture a spanning text while the former cancel this shrinkage. The macros `\pcol@output` and `\pcol@output@start` also refer to this register to let `\vsize` have it in the former and to calculate the room for each column-page in the starting page in the latter. The macro `\pcol@output@end` lets the register have `\textheight` for the post-environment stuff because the column-pages above it simply precedes the stuff in the main vertical list. The other users `\pcol@makefcolumn` and `\pcol@makefcolelt` use this register to accumulate the total size of floats to be put in a float column temporarily.

`\@pageht` is a `\dimen` register to be used in L^AT_EX’s `\@specialoutput` to have the height of `\@holdpg`. It is referred to by `\pcol@addmarginpar` to determine the position at which a marginal note is placed. We also use it as a scratchpad to have V_P in `\pcol@sync` and `\pcol@measurecolumn`, and to save $\pi^h(p_t)$ in `\pcol@flushcolumn` for the reference in itself, and to do so in `\pcol@makeflushedpage` for `\pcol@imakeflushedpage`.

`\@pagedp` is a `\dimen` register to be used in L^AT_EX’s `\@specialoutput` to have the depth of `\@holdpg`. However, we use it as a scratchpad in `\pcol@sync` and `\pcol@measurecolumn` to have D_P , and in `\pcol@output@end` to have the value to be set in `\pcol@prevdepth`.

`\@toproom` is a `\dimen` register to have the room for top floats. The register is saved in $\kappa_c(\rho_t)$ by `\pcol@setcurrcol` and restored from it by `\pcol@iigetcurrcol`. The macro `\pcol@makefcolumn` uses this register as a flag to indicate that $\kappa_c(\lambda_t)$ of the column c having $\kappa_c(\rho_t) = \infty$ contains floats to be put in its last float column possibly as top floats so that it is examined by `\pcol@flushcolumn` and `\pcol@imakeflushedpage`, the former of which then lets $\kappa_c(\rho_t) = 0$ to mean the floats are put in a float column in a non-last page by the macro.

`\@botroom` is a `\dimen` register to have the room for bottom floats. The register is saved in $\kappa_c(\rho_b)$ by `\pcol@setcurrcol` and restored from it by `\pcol@iigetcurrcol`.

`\@fpmn` is a `\dimen` register to have `\floatpagefraction` \times `\@colht` being the minimum total size of floats for which an ordinary (not flushed) float column can be build. It is referred

to by `\pcol@makefcolumn` as the threshold below which floats in the last float column can be put as top floats.

`\mparbottom` is a `\dimen` register to have the bottom position of the last `\marginpar` stuff. Its value at `\begin{paracol}` is referred to by `\pcol@output@start` to let M_L^l or M_L^r of $\pi^m(0)$ has an element based on it, while the tail of one of the lists in $\pi^m(p_t)$ defines the value at `\end{paracol}` which `\pcol@output@end` lets the register have. The register is also updated by `\pcol@getmparbottom` and `\pcol@getmpbelt` to let `\pcol@addmarginpar` being L^AT_EX's original `\@addmarginpar` know the uppermost available position for the marginal note to be added. This update is, however, just for communication between these macros and thus is ineffective for typesetting posterior to that, as well as the update in `\pcol@addmarginpar`, because whole information for marginal note placement is kept in $\pi^m(p)$ in Π^+ .

`\textfloatsheight` is a `\dimen` register to have the combined height of mid floats and their separators. It is initialize to be 0 by `\pcol@floatplacement`, saved in $\kappa_c(\eta)$ by `\pcol@setcurrcol`, and restored from it by `\pcol@iigetcurrcol`.

`\dimen@` is a `\dimen` register for temporary use. It is used in the following macros.

- `\pcol@buildcolseprule`, `\pcol@buildcselet@S` and `\pcol@buildcselet` to have the argument $d \in \{\@maxdepth, 0\}$ of the first macro.
- `\pcol@bg@paintregion@i` to have y_1 of $R_a^{[c]}$.
- `\pcol@bias@mpbout@i{y}{t}{b}` to have t and then $t + y$.
- `\pcol@output@switch` to have the height of pre-spanning-text stuff in `\pcol@prespan`, or 0 if it is \perp .
- `\pcol@sync` to have V or $V - D_T + V_E$.
- `\pcol@addflhd` and `\pcol@hdflelt` to measure the height of top and bottom floats, `\pcol@makecol` and `\pcol@output@switch` to measure the height of pre-spanning-text stuff including the top floats, and `\pcol@measurecolumn` for top and bottom floats and V_T , V_B and V_P .
- `\pcol@setcolwidth@s` and `\pcol@setcw@accumwd` to accumulate W being the sum of natural widths of column/gap specifications, and then used by `\pcol@setcw@calc factors` to calculate W/W_T and $W - W_T$.

`\dimen@ii` is a `\dimen` register for temporary use. It is used in the following macros.

- `\pcol@makecol` to measure the total height of top floats by `\pcol@addflhd`.
- `\pcol@bg@addext` to have $e = \text{pcol@bg@ext} \cdot d \cdot \{a \cdot c, a\}$ and then $10000 \text{ pt} - e$ to calculate an extension of background painting.
- `\pcol@bias@mpbout@i{y}{t}{b}` to have b and then $b + y$.
- `\pcol@measurecolumn` to measure V_T , V_P and D_P .
- `\pcol@setcolwidth@s` and `\pcol@setcw@accumwd` to accumulate F being the sum of infinite stretch factors in column/gap specifications with the unit of pt, and then used by `\pcol@setcw@calc factors` to calculate $(W - W_T)/F$, where W is the sum of natural widths.
- `\pcol@setcw@calc factors` to have $(W - W_T)/F$ above or 0 to be used in `\pcol@def@extract@fil@iii` through `\pcol@setcw@filunit` made `\let-equal` to the register by `\pcol@setcolwidth@s`.

`\@tempdima` is a `\dimen` register for temporary use. The usages of this register are as follows.

- In `\pcol@makecol` and `\pcol@startpage`, it is used to throw $\pi^h(p_t)$ away when we get it by `\pcol@getcurrpinfo` because we don't need it.
- In `\pcol@outputelt`, it has $\pi^h(p)$ to examine if p is a float page.
- In `\pcol@ioutputelt`, it has $\pi^h(p)$ possibly shrunk by page-wise footnotes to know the backgrounds to be painted for columns etc. After that it has w_c being the width of each `\hbox` into which the column-page of each column c is put.
- In `\pcol@buildcolseprule` and its callees `\pcol@buildcselet@S` and `\pcol@buildcselet`, the register has the first argument $H = \pi^h(p)$ of the caller macro.
- In `\pcol@hfil⟨c⟩`, it has $g_c = \pcol@columnsep \cdot c$.
- In `\@outputpage`, it has the sum of `\topmargin`, `\headheight` and `\headsep` being the distance between tops of imaginary page and its text area.
- In `\pcol@startcolumn`, it is used to save `\@colht` which can be shrunk temporarily by page-wise footnotes.
- In `\pcol@bg@paintregion@i`, it is let have x_0 of $R_a^{[c]}$.
- In `\pcol@output@start`, it is used to have the room for each column-page in the starting page, and then the height-plus-depth of the pre-environment stuff.
- In `\pcol@output@switch`, it is used to throw $\pi^h(p)$ away when we get it by `\pcol@getcurrpinfo` because we don't need it.
- In `\pcol@shiftspanning`, it is let have the left-shift amount of a spanning text in column-swapping.
- In `\pcol@restartcolumn`, it is used to save `\@colht` which can be shrunk temporarily by page-wise footnotes.
- In `\pcol@unvbox@ccclv⟨ins⟩`, it has the depth of `\box255` for going back to the baseline of the box, and then has the natural component of `\skip⟨ins⟩` to add its stretch and shrink components only.
- In `\pcol@addmarginpar`, at first it has the distance between left edges of the leftmost and current columns. Then it is let have the distance between top edges of the column and the marginal text to be put.
- In `\pcol@getmparbottom⟨t⟩⟨h⟩` and `\pcol@getmpbelt⟨t_i⟩⟨b_i⟩`, it at first has t and then is let have b_i when the marginal note cannot be put at t .
- In `\pcol@sync`, `\pcol@measurecolumn` and `\pcol@synccolumn`, it has V_T being the maximum combined height of top floats and the main vertical list.
- In `\pcol@makefcolumn` and `\pcol@makefcolelt`, it has the room for floats to be put in a float column.
- In `\pcol@makeflushedpage`, it has the height-plus-depth of spanning stuffin $\pi^i(p_t)$.
- In `\pcol@output@end`, at first it is let have $V'_p - H$, where H is the height(-plus-depth) of `\@outputbox` having the ship-out image of the last page, being the negative counterpart of the height-plus-depth of spanning stuff in the last page for setting \mathcal{M} , and then have H to be set in `\pcol@bg@preposttop` for the background painting of post-environment stuff.
- In `\pcol@setcolwidth@r`, it has $\text{textwidth} - (C^1 - C^0 - 1)\text{columnsep}$ being the base of w_c .

- In `\pcol@setcw@getspec@i`, it is let have the natural width of a column/gap specification, to be used in `\pcol@setcolwidth@s`, `\pcol@setcw@accumwd` and `\pcol@setcw@set`, while in the last of them it finally has w_c or g_c .
- In `\pcol@setcw@calcf⟨x⟩⟨y⟩⟨z⟩`, at first it has y , then $y/2^{k_2}$, and then $\lceil y/2^{k_2+k_3} \rceil$, where k_2 and k_3 are scaling parameters for good approximation.
- In `\pcol@switchcol`, it is let have what `\pcol@ensurevspace` has so that the dimensional expression in it is evaluated in the macro and the evaluation result is assigned to $V_E = \pcol@@ensurevspace$.
- In `\ensurevspace{space}`, it is let have $\langle space \rangle$ to ensure the argument is a dimension including forced one.
- In `\pcol@fntextbody`, it has the height-plus-depth of the `\vbox` in which the footnote is encapsulated.
- In `\pcol@set@color@push`, it has the width of the `\vbox` to be `\inserted`, which is $m\text{ sp}$ for a math-mode push of $\gamma_{i,m}$ or 0 for a non-math one γ_i .
- In `\pcol@bg@defext{d}{e}`, it is let have e to confirm e is a proper dimension.

`\@tempdimb` is a `\dimen` register for temporary use. The usages of this register are as follows.

- In `\pcol@makecol`, it is used to measure the height-plus-depth h_i of decapsulated `\box255` and its original form to add an element $\text{span}(H_i, h_i)$ to $\pi^s(p_t)$ for a spanning text captured in the box.
- In `\pcol@ioutputelt`, it has the height-plus-depth of spanning stuff $\pi^b(q)$ to be temporarily added to `\topmargin`.
- In `\pcol@buildcolseprule` it has H_0+h_0 , while in its callee `\pcol@buildcselt⟨H_i⟩⟨h_i⟩` it has $H_{i-1}+h_{i-1}$ and then H_i+h_i where $\text{span}(H_i, h_i) \in \pi^s(p)$.
- In `\pcol@bg@paintregion@i`, it is let have y_0 of $R_a^{[c]}$.
- In `\pcol@output@switch`, it is let have the height-plus-depth h_i of `\@holdpg` having a spanning text to add an element $\text{span}(H_i, h_i)$ to $\pi^s(p_t)$.
- In `\pcol@shrinkcolbyfn`, it is let have the inverse of the `\skip` component of the argument `\insert` register of the macro, so that in `\pcol@startcolumn` and `\pcol@restartcolumn` it has that of $\pi^f(p)$ if p has page-wise footnotes, or 0 otherwise, and then is referred to by `\pcol@deferredfootins` which then lets the register have the height cap of Φ splitting.
- In `\pcol@addmarginpar`, `\pcol@getmparbottom` and `\pcol@getmpbelt`, it is let have the vertical space to be occupied by the marginal text to be put, being the second argument of `\pcol@getmparbottom`.
- In `\pcol@sync` and `\pcol@measurecolumn`, it has V_B and then, in the former, it has $V_P+v(f)$, V_T or $V_T+V_B+v(f)$ according to the contents of the page to be synchronized.
- In `\pcol@makefolelt`, it has the size of vertical space consumed by a float.
- In `\pcol@synccolumn`, it has $V_T-v_c(t)$ being the vertical space from the bottom of $\kappa_c(\beta^b)$ to the synchronization point. If the synchronization point is defined by a column without main vertical list but with top floats, then the register is let have $V_T-v_c(t)+\text{textfloatsep}-\text{floatsep}+10000\text{pt}$ to be set in $\kappa_c(\xi) = \pcol@textfloatsep$ as the space below the MVL-float biased by 10000pt to indicate the last float is the MVL-float.

- In `\pcol@setcolwidth@r`, it has $\text{\textwidth} - (C^1 - C^0 - 1)\text{\columnsep} - \sum_{d=0}^{k-1} w_d$ being the base of w_c for $c \in [C^0 + k, C^1)$ where k is the number of fractions given in the argument of `\columnratio`.
- In `\pcol@setcw@calcf⟨x⟩⟨y⟩⟨z⟩`, at first it is let have x , then $x \cdot 2^{k_1}$, then $z' = \lfloor (x \cdot 2^{k_1}) / \lceil y / 2^{k_2 + k_3} \rceil \rfloor$, and finally $Z = z \times 1\text{pt} = z' \cdot 2^{16-k}$ referred to by `\pcol@setcw@calcfactors` as $\phi_f = (W_T - W)/F$, where k_1, k_2 and k_3 are scaling parameters for good approximation and $k = k_1 + k_2 + k_3$.
- In `\pcol@extract@fil@ii` and `\pcol@extract@fil@iii`, it is let have the infinite stretch factor of a column/gap specification represented with the unit `\pcol@setcw@filunit`, to be used in `\pcol@setcolwidth@s`, `\pcol@setcw@accumwd`, and `\pcol@setcw@set`.
- In `\pcol@fntextbody`, it has $\text{\textheight} - \text{\skip}\text{\footins}$ as the cap of the footnote.

`\@tempdimc` is a `\dimen` register for temporary use. It is let have values as follows.

- $H_i - (H_{i-1} + h_{i-1})$ in `\pcol@buildcset⟨H_i⟩⟨h_i⟩`.
- x_1 of $R_a^{[c]}$ in `\pcol@bg@paintregion@i`.
- $t + h$ in `\pcol@getmparbottom⟨t⟩⟨h⟩`.
- $\max(t, b_{i-1}) + h$ in `\pcol@getmpbelt⟨t_i⟩⟨b_i⟩` invoked from `\pcol@getmparbottom⟨t⟩⟨h⟩`.
- D_T in `\pcol@sync`, `\pcol@measurecolumn` and `\pcol@synccolumn`.
- `\floatsep` or `\@fpsep` in `\pcol@makefcolumn` and `\pcol@makefcolelt`.
- w_c being the width of column c in `\pcol@setcolwidth@r`.
- $W_T - W$ in `\pcol@setcw@calcfactors`.
- At first for calculation of $y/2^{k_2}$ and then $z'/2^k \approx x/y$ in `\pcol@setcw@calcf⟨x⟩⟨y⟩⟨z⟩` where k_2 and k are scaling parameters for good approximation.

2.1.4 Skip Registers

`\baselineskip` is \TeX 's primitive register to have the vertical skip to separate adjacent baselines. It is referred to by `\pcol@output` and `\pcol@output@start` to examine if `\@colroom` is unexpectedly small, and by `\pcol@switchcol` to give it to `\ensurevspace` to let `\pcol@ensurevspace` have the default value.

`\topskip` is \TeX 's primitive register to have the vertical skip from the top edge of a page to the baseline of its first vertical item. It is let be 0 by `\pcol@output@start` if we have pre-environment stuff and is saved in $\pi^t(0)$, while `\pcol@startpage` lets it be `\pcol@topskip`, into which `\pcol@zparacol` saves the value outside `paracol` environment, saving the value in $\pi^t(p)$. Then the register is restored from $\pi^t(p)$ by `\pcol@getpelt` and `\pcol@sync`, while `\pcol@synccolumn` refers to the value restored by the latter to adjust a synchronization point. The macro `\pcol@putbackmvl` lets the register have 0 when it starts a spanning text because it originally follows pre-spanning-text stuff in the column-page to be restarted rather than at the page top. The macro `\pcol@output@end` temporarily lets the register have 0 if we have non-empty columns in the last page, while `\endparacol` restores it from `\pcol@topskip` for the pages outside `paracol` environment.

`\splittopskip` is TeX's primitive register to have the vertical skip inserted at the beginning of the second half of the box split by `\vsplit` or TeX's internal operation for splitting an `\insert` at a page break. The register is temporarily let have 0 by `\pcol@deferredfootins` when it splits Φ so that the second half does not have any skip at the top. The register is also let have `\footnotesep` in `\pcol@fntextbody` for footnote typesetting.

`\parskip` is TeX's primitive register to have the vertical skip inserted above each paragraph. It is referred to by `\pcol@zparacol` to nullify the insertion going to be made by the first `\item` of a list-like environment, when the macro finds the `paracol` environment to start is at the very beginning of a list-like environment.

`\fill` is an API `\skip` register to have a skip `0pt plus 1fill`. In our macros, however, it is used as a keyword in `\pcol@setcw@getspec`, `\pcol@setcw@getspec@i` and `\pcol@setcw@fill` to extract the infinite stretch factor f given as `f\fill` in the specification.

`\itemsep` is an API `\skip` register to have the vertical skip inserted above each non-first `\item` in list-like environments. It is referred to by `\pcol@zparacol` to nullify the insertion going to be made by the first `\item` of a list-like environment, when the macro finds the `paracol` environment to start is at the very beginning of a list-like environment.

`\floatsep` is an API `\skip` register to have the vertical skip between adjacent floats. It is referred to by `\pcol@cflt` to cancel the skip following the last float, by `\pcol@makefcolum` to let `\pcol@makefcolelt` examine the capacity of a float column in the last page, by `\pcol@addflhd` and `\pcol@hdfilelt` to measure the total height of top and bottom floats, and by `\pcol@sync` to calculate the space below the MVL-float.

`\textfloatsep` is an API `\skip` register to have the vertical skip between the main vertical list and top/bottom floats. It is referred to by `\pcol@output@start` to calculate the room for each column-page in the starting page with bottom floats in the pre-environment stuff, by `\pcol@combinefloats` to insert a skip below the bottom floats in the pre-environment stuff and last page, by `\pcol@measurecolumn` to take this skip into account in the calculation of V'_p , by `\pcol@addflhd` to measure the vertical space for top and bottom floats, and by `\pcol@synccolumn` to calculate the synchronization point for columns with top floats.

`\dblfloatsep` is an API `\skip` register to have the vertical skip between adjacent page-wise floats, and is used in `\pcol@startpage` to cancel the skip below the last float.

`\dbltextfloatsep` is an API `\skip` register to have the vertical skip between the last page-wise float and the top of columns, and is used in `\pcol@startpage` to put the skip.

`\@topsep` is a `\skip` register to have the vertical skip inserted above the first `\item` of a list-like environment. The actual value is determined by `\@trivlist` from API parameters `\topsep`, `\partopsep` and `\parskip` depending on how the environment appears. The skip in the register is inserted by `\pcol@zparacol` when it finds the `paracol` environment to start is at the very beginning of a list-like environment.

`\@fptop` is a `\skip` register to have the vertical skip inserted at the top of a float column, and is used in `\pcol@makefcolpage`.

`\@fpsep` is a `\skip` register to have the vertical skip between adjacent floats in a float column, and is used in `\pcol@makefcolpage`.

`\@fpbot` is a `\skip` register to have the vertical skip inserted at the bottom of a float column, and is used in `\pcol@makefcolpage`.

`\@tempkipa` is a `\skip` register for temporary use. It is used in the following macros.

- `\pcol@makecol`, `\pcol@startpage`, `\pcol@outputelt`, `\pcol@output@switch`, `\pcol@flushcolumn` and `\pcol@makeflushedpage` to throw $\pi^t(p)$ away because we don't need it.
- `\pcol@output@start` to determine `\topskip` for the starting page.
- `\pcol@setcw@getspec@i` and `\pcol@setcw@fill` to have the width specification of a column or gap.

It is also used in the top level invocation of `\pcol@defkw` with a glue of `0pt plus 1fil minus 1fil`.

2.1.5 Box Registers

`\strutbox` is an API `\box` register to have the strut for the current font size. It is used in `\pcol@fntextbody{text}` to let `\splitmaxdepth` have its depth, and to let the last line of the footnote `\text` have its depth at shallowest.

`\@cclv` is a `\box` register but T_EX defines that it has the main vertical list when `\output` routine is invoked. It is referred to by `\pcol@makecol` when it has a broken spanning text to measure its height-plus-depth for the element to be added to $\pi^s(p_t)$ and to update it combining its contents with pre-spanning-text stuff optionally shifting it left by passing the register to `\pcol@shiftspanning`. The macro also uses the register together with its callee `\pcol@unvbox@cclv` to add stretch/shrink factor of `\skip\footins` at its bottom for a column-page in a page having page-wise footnotes. The macro `\pcol@specialoutput` examines the register to discard the dummy `\vbox` inserted in it by `\pcol@invokeoutput`. The other users `\pcol@output@start`, `\pcol@makenormalcol`, `\pcol@flushcolumn` and `\pcol@imakeflushedpage` let the register have the main vertical list of pre-environment stuff or a column to be passed to `\@makecol`, and `\pcol@flushcolumn` also takes care the skip above page-wise footnotes.

`\voidb@x` is a `\box` register to be void (\perp) always. It is used to initialize `\pcol@prespan` and `\pcol@rightpage` at their declaration, and is referred to by the following macros.

- `\pcol@makecol` to make `\pcol@currfoot` void unless page-wise footnotes in `\footins` is saved into $\pi^f(p)$.
- `\pcol@startpage` to let $\pi^i(p_t) = \perp$ if the new top page does not have spanning stuff and $\pi^f(p_t) = \perp$ for all float pages and the new top page.
- `\pcol@outputelt` to initialize `\@outputbox`.
- `\pcol@ioutputelt` to examine if S_c is empty.
- `\pcol@output@start` to let $\pi^f(0) = \perp$, and $\gamma_0^c = \perp$ if $\hat{\gamma}_0^c$ is undefined.
- `\pcol@output@switch` to let $\kappa_c(\tau^b) = \perp$ if the column does not have column-wise footnotes.
- `\pcol@getcurrfoot` to let `\footins` be void if so.
- `\pcol@setcurrcolnf` to let $\kappa_c(\tau^b) = \perp$ because the column c does not have column-wise footnotes.

- `\pcol@putbackmvl` to let `\pcol@prespan = ⊥` if a spanning text really starts from the top of a column-page, and $\Gamma_s = \perp$ if the column-page $\kappa_c(\beta)$ to be restarted is non-empty.
- `\pcol@savestack` to let $\Gamma_s = \perp$ or its first item be \perp if Γ^c or γ_0^c is \perp , respectively.
- `\pcol@savefootins` to let its argument macro have a void box if `\@freelist` is exhausted.
- `\pcol@makeflushedpage` to initialize `\@outputbox` and `\pcol@rightpage` to be \perp if the flushed page does not have spanning stuff, and to let $\pi^f(p_t) = \perp$ after putting it in the last page so that `\pcol@output@end` will be unaware of the page-wise and non-merged footnotes.
- `\pcol@flushfloats` to let `\pcol@rightpage = ⊥` if parallel-paging is not in effect.
- `\pcol@output@end` to let `\pcol@rightpage = ⊥` if the last page has nothing other than spanning stuff being page-wise floats and thus we don't have the right parallel-page. The macro also lets $\gamma_0^c = \perp$ for all c and $\Gamma = \perp$.
- `\pcol@com@flushpage` and `\pcol@com@clearpage` gives the void box to `\pcol@flushclear` as its argument to mean these macros are only aware of `\ifpcol@flush` as the result of pre-flushing column height check. The macro `\endparacol` also does that if the footnote typesetting is merged.

`\@holdpg` is a `\box` register to have the main vertical list when `\output` is invoked with a special `\penalty` code. It is let have that by `\pcol@specialoutput`, and is referred to by `\pcol@output@start` and `\pcol@makenormalcol` for pre-environment stuff, and by `\pcol@output@switch` for the column from which we are leaving.

`\@outputbox` is a `\box` register to have a partial or the complete ship-out image of a column or a page. The usages of the register are as follows.

- In `\pcol@@makecol`, it has a column-page made by `\@makecol` for `\pcol@flush` column and `\pcol@imakeflushedpage`.
- In `\pcol@combinefloats`, it has a column-page to which top and bottom floats are combined.
- In `\pcol@cf1t`, it has a column-page to which top floats are combined.
- In `\pcol@opcol`, it has the complete column-page built by `\@makecol`.
- In `\pcol@startpage`, it has the complete float page built by `\@tryfcolumn`.
- In `\pcol@outputelt`, it has the complete (left parallel-) page to be shipped out by `\@outputpage`.
- In `\pcol@outputpage@r`, it is temporarily made `\let`-equal to `\pcol@rightpage` so that the box is shipped out by `\pcol@@outputpage` being L^AT_EX's `\@outputpage` instead of the real `\@outputbox`.
- In `\pcol@output@start`, it has the pre-environment stuff built by `\pcol@makenormalcol`.
- In `\pcol@combinefootins`, it is let have the pre-environment stuff with footnotes.
- In `\pcol@flushcolumn`, it has a flushed column-page built by `\@makecol` or a float column built by `\@makefcolumn`.

- In `\pcol@output@flush` and `\pcol@output@clear`, it has a flushed page built by `\pcol@makeflushedpage` and `\pcol@imakeflushedpage` in which it has each of flushed column-page built by `\@makecol`.
- In `\pcol@flushfloats`, it has the complete (left parallel) page for flushed float columns.
- In `\pcol@iflushfloats`, it has a float column built by `\pcol@makefcolumn`.
- In `\pcol@output@end`, it has the ship-out image of the last page of a `paracol` environment built by `\pcol@makeflushedpage` and `\pcol@imakeflushedpage`.

`\@tempboxa` is a `\box` register for temporary use. The usages of the register are as follows.

- In `\pcol@makecol`, it is used to decapsulate `\box255` containing a broken spanning text. In the macro and `\pcol@output@switch`, it is also used as a waste basket to make `\footins` void when it contains page-wise footnotes in a non-top page.
- In `\pcol@cflt` and `\pcol@startpage`, it has top column-/page-wise floats combined by the application of `\@comflelt/\@comdblfllelt` to `\@toplist/\@dbltoplist` respectively.
- In `\pcol@phantom⟨b⟩`, it has an empty box whose height and depth are equal to those of the argument box b .
- By `\pcol@buildcolseprule` and its callees `\pcol@buildcselt@S` and `\pcol@buildcselt`, it is let have the painted backgrounds for columns, column-separating gaps and spanning texts in a page, and then is put into the ship-out image of the page by `\pcol@ioutputelt`, `\pcol@imakeflushedpage` or `\pcol@iflushfloats`.
- In `\@outputpage`, it is let have the painted background of the right page referred to by its callee `\pcol@outputpage@r`.
- In `\pcol@bg@paint@i⟨body⟩`, it is let have painted backgrounds built by $\langle body \rangle$.
- In `\pcol@bg@paintregion⟨a⟩⟨c⟩`, it is let have painted background of the region $R_a^{[c]}$.
- In `\pcol@specialoutput`, it is used to discard the dummy `\vbox` put by `\pcol@invokeoutput`.
- In `\pcol@makenormalcol`, it is used to save `\footins` into it and make it \perp temporarily to exclude merged footnotes from spanning stuff for pre-environment stuff.
- In `\pcol@ifempty⟨box⟩⟨then⟩⟨else⟩`, it is used to examine if $\langle box \rangle$ is empty.
- In `\pcol@scancst⟨box⟩` and `\pcol@iscancst`, it is used to have what Γ or Γ_s has after the scan of $\langle box \rangle \in \{\Gamma, \Gamma_s\}$.
- In `\pcol@savecolorstack`, it is used to have the `\vbox` for γ_0^c to be placed at the top of Γ_s .
- In `\pcol@deferredfootins`, it is used to have the first half split from Φ being the deferred footnotes to be `\inserted`.
- In `\pcol@fntextbody⟨text⟩`, it is used to encapsulate $\langle text \rangle$ in it.
- In `\pcol@icolumncolor`, it is used to have a `\vbox` to be `\inserted` for the update of γ_0^c .
- In `\pcol@set@color@push`, it is used to have a `\vbox` to be `\inserted` to push γ_i or $\gamma_{i,m}$ to Γ_r .
- In `\pcol@reset@color@pop` and `\pcol@reset@color@mpop`, it is used to have a `\vbox` to be `\inserted` to add γ_i^- or $\gamma_{i,m}^-$ to Γ_r .

2.1.6 Token Registers

`\output` is \TeX 's primitive to have `\output` routine. It is let have `\pcol@output` as its sole token by `\pcol@zparacol`.

`\everypar` is \TeX 's primitive to have tokens inserted at the beginning of each paragraph. In `\pcol@sptext` and `\pcol@com@endcolumn`, it is `\globalized` to keep its contents after the end of a group. In `\pcol@output@switch`, its contents are broadcasted to $\kappa_c(\varepsilon)$ for all $c \in [0, C)$ if columns are synchronized with a spanning text. Then these values or that simply given in a column are saved into $\kappa_c(\varepsilon)$ by `\pcol@setcurrcol`, and then restored from it by `\pcol@iigetcurrcol`.

`\everyvbox` is \TeX 's primitive to have tokens inserted at the beginning of each `\vbox`. In `\pcol@zparacol`, after tokens in it are saved into `\pcol@everyvbox`, it is let have a `\the`-reference to `\pcol@everyvbox` and `\pcol@innertrue` to turn `\ifpcol@inner = true`, and then the register itself is made `\let`-equal to `\pcol@everyvbox`. In addition, it is let have tokens in `\pcol@everyvbox` if a `\global` assignment to the register is made in the `paracol` just having been closed. Another usage of this register is to insert a painted page background to the `\vbox` to be `\shipout` by `\pcol@@outputpage` being \LaTeX 's `\@outputpage`, and is used for this purpose by `\pcol@outputpage@l` and `\pcol@outputpage@r`, and by `\pcol@outputpage@ev` to nullify this special function for other inside `\vboxes`.

`\@temptokena` is a `\toks` register for temporary use. It is used in `\pcol@output@switch` to broadcast `\everypar` to $\kappa_c(\varepsilon)$ for all $c \in [0, C)$.

2.1.7 Switches

`\if@twocolumn` is a switch to be *true* iff multi-column pages are being typeset. It is turned *true* by `\pcol@zparacol`, and then turned *false* by `\endparacol`. In addition, it is turned *false* when `\pcol@output` finds that the `\output` request for a page break outside `paracol` is sneaked into our own `\output` routine, in order to avoid that \LaTeX 's original `\output` routine misunderstands it is working on a two-columned document. The switch is examined by \LaTeX 's own macros including old `\end@dblfloat` kept in our own `\pcol@end@dblfloat`. It is also examined by `\pcol@zparacol` before being turned *true* to ensure it is *false* or to complain about the inappropriateness otherwise.

`\if@firstcolumn` is a switch to be *true* iff the first column is being typeset. Its truth value is determined by `\pcol@addmarginpar` to tell `\pcol@@addmarginpar`, `\pcol@getmparbottom@i` and `\pcol@setmpbelt@i` the margin which a marginal note goes to.

`\if@twoside` is a switch to be *true* iff two-sided page typesetting is in effect and thus even numbered page may have their own left margins, headers and footers different from those for odd numbered pages. Besides the initialization by the main class file such as `article.cls` according to the class option `twoside`, the switch is `\globally` turned *false* by `\pcol@twosided` for the case in which API macro `\twosided` does not have 'p' in its optional argument, and then `\globally` turned *true* by `\pcol@twosided@p` which is invoked when the argument contains 'p', or the API macro does not have the argument at all. Then the switch is referred to by `\pcol@outputpage@l`, `\pcol@outputpage@r` and `\pcol@bg@swappage` to decide the left margin of even numbered pages, i.e., `\evensidemargin` if the switch is *true* or `\oddsidemargin` otherwise. The switch is also referred to by `\pcol@com@cleardoublepage` to have a blank page if the switch is *true* and the command `\cleardoublepage` is used in an odd-numbered page.

`\if@reversemargin` is a switch to be *true* iff `\reversemarginpar` is specified to reverse the side which marginal notes go to. It is examined by `\pcol@addmarginpar` as a factor to decide the margin which a marginal note goes to, and by `\pcol@do@mpbout@i` for the same purpose but for marginal notes in pre-environment or post-environment stuff.

`\if@mparswitch` is a switch to be *true* iff it is specified by, for example, `twoside` option of a class such as `article`, that marginal notes in even numbered pages go to the left margin. It is examined by `\pcol@do@mpbout@i` as a factor to decide the margin which a marginal note goes to in pre-environment or post-environment stuff.

`\if@nobreak` is a switch to be *true* iff the last paragraph is for a sectioning command. The switch is saved into $\kappa_c(\sigma)$ together with `\if@afterindent` by `\pcol@setcurrcol`, and then restored from it by `\pcol@iigetcurrcol`. The macro `\pcol@output@switch` refers to it to broadcast its value set by a spanning text to $\kappa_c(\sigma)$ for all $c \in [0, C)$, while `\pcol@output@start` and `\pcol@restartcolumn` insert `\penalty = 10000` by `\nobreak` if the switch is *true*. This conditional `\nobreak` is also done by `\pcol@icolumncolor`, `\pcol@set@color@push`, `\pcol@reset@color@pop` and `\pcol@reset@color@mpop` to avoid a break after an `\insert`. The macro `\pcol@zparacol` also examines the switch, but with the truth value in it given outside `paracol` environment, to invoke `\@nbitem` if *true* when the macro finds the `paracol` environment to start is at the very beginning of a list-like environment.

`\if@newlist` is a switch to be *true* in the duration after a list-like environment starts and until its first `\item` appears. The switch is examined by `\pcol@zparacol` to know if the `paracol` environment to start is at the very beginning of a list-like environment and, if so, is turned *false* by the macro after it inserts vertical skips pretending the first `\item` is given.

`\if@inlabel` is a switch to be *true* in the duration after an `\item` appears and until its first paragraph is given. The switch is examined by `\pcol@zparacol` together with `\if@newlist` to know if the `paracol` environment to start is at the very beginning of a list-like environment (*false*) and not `trivlist`-like one (*true*).

`\if@afterindent` is a switch to be *true* iff a sectioning command tells that the first paragraph following it is to be indented. The switch is saved into $\kappa_c(\sigma)$ together with `\if@nobreak` by `\pcol@setcurrcol`, and then restored from it by `\pcol@iigetcurrcol`. The macro `\pcol@output@switch` refers to it to broadcast its value set by a spanning text to $\kappa_c(\sigma)$ for all $c \in [0, C)$.

`\if@fcolmade` is a switch to be *true* iff a float column or float page is built by `\@tryfcolumn` or `\@makefcolumn`. The value is set by `\@tryfcolumn` for $\kappa_c(\lambda_d)$ is referred to by `\pcol@output`, `\pcol@startcolumn` and `\pcol@freshpage`, while that for `\@dbldeferlist` is referred to by `\pcol@startpage`. The value set by `\@makefcolumn` for $\kappa_c(\lambda_d)$ is referred to by `\pcol@flushcolumn`, while that for `\@dbldeferlist` is referred to by `\pcol@output@clear`. The macros `\pcol@flushfloats` and `\pcol@iflushfloats` also refer to the switch to build pages having only float columns and turn the switch *true* or *false* by themselves to know such pages are still to be built or not. The macro `\pcol@output@end` also turns the switch *true* if a last page will be followed by page(s) having float columns to tell that to `\pcol@flushfloats`.

`\if@tempswa` is a switch for temporary use. The usages of the switch are as follows.

- In `\pcol@checkshipped`, it is turned *true* iff S_c for all $c \in [0, C)$ have column-pages to be shipped out, and then it is examined by `\pcol@opcol`.
- In `\pcol@nextpage` and `\pcol@nextpelt`, it is *true* until `\pcol@nextpelt` finds the first q such that $q > p$ and $\pi^h(q) \geq 0$ to mean q is not for a float page, so that we let $p = q$ to skip float pages following to the old p if any.
- In `\pcol@outputcolumns` and `\pcol@outputelt`, it is *true* until `\pcol@outputelt` finds the first q such that $q \geq p_b$ and $\pi^h(q) \geq 0$ to mean q is not for a float page, and the argument of `\pcol@outputcolumns` is 0 to mean that it is not for page flushing, so that we ship out q and all float pages following it if any.
- In `\@outputpage` it is let have the value of `\ifpcol@bg@painted` indicating if background painting for the left page is done, and then it is examined by `\pcol@outputpage@l` to determine whether the background is put into the final ship-out image.
- In `\pcol@makenormalcol`, it is *true* iff the footnotes in pre-environment stuff is included in `\@outputbox` which the macro builds.
- In `\pcol@output@switch`, at first it holds `\if@nobreak` of the spanning text if columns are synchronized with it to broadcast `\if@nobreak` to all $\kappa_c(\sigma)$. Then it is turned *true* iff `\ifpcol@sync = true` for synchronization or `\ifpcol@clear = true` for flushing, so as to invoke `\pcol@sync`. And finally, it is turned *true* iff `\ifpcol@clear = false` or `\ifpcol@sync = true`, so as to invoke `\pcol@restartcolumn`.
- In `\pcol@restartcolumn`, it is turned *true* iff footnote typesetting is page-wise and $p < p_t$.
- In `\pcol@scancst`, it is initialized to be *true*. Then it is referred to by `\pcol@iscancst` for each $\gamma \in \Gamma_r$ to update γ_0^c and then turned *false* when the first one is found.
- In `\pcol@savecolorstack`, it is *true* iff either $\Gamma \neq \perp$ or $\gamma_0^c \neq \perp$, i.e., Γ^c to be saved is not \perp .
- In `\pcol@getmparbottom`, it is initialized to be *false* and then may be turned *true* by `\pcol@getmpbelt` if it finds a gap between two marginal notes to accommodate that to be added, and then examined by `\pcol@getmparbottom` to know the fact.
- In `\pcol@sync`, it is turned *true* iff the synchronized or flushed page can be built by `\pcol@synccolumn`.
- In `\pcol@makefcolumn` having non-empty $\kappa_c(\lambda_t)$, it is turned *false* iff the macro is acting on a column in the last page, $\kappa_c(\lambda_d)$ is emptied by the macro itself, and the total size of the floats to be put in the float column being built by the macro is less than `\@fpm`, to mean it is possible that the floats in $\kappa_c(\lambda_t)$ is put in the float column as top floats.
- In `\pcol@measurecolumn` and `\pcol@addflhd`, it is set to be *false* iff both top floats and the main vertical list are empty, so that `\pcol@measureupdate` examines it for the update of V_T and D_T . Then it is kept *false* iff both of footnotes and bottom floats are empty, so that `\pcol@measurecolumn` examines it for the update of V_P and `\pcol@measureupdate` does for V'_P and D_P .
- In `\pcol@makeflushedpage`, it is made *false* iff p_t is the last page, $V'_P = -\infty$ to mean all columns are empty and $\pi^f(p_t) = \perp$, so as to make the spanning stuff in $\pi^i(p_t)$ a float in post-environment stuff if `\ifpcol@dfloats` also *false*. Then it is

kept *false* if `\ifpcol@dfloats = false` or $\pi^i(p_t) = \perp$ to mean nothing is shipped out for last page. Then it is made *false* iff p_t is the last page without deferred floats and merged footnote typesetting is in effect, i.e., the switch is *true* iff page-wise footnotes are put in the page to be flushed.

- In `\pcol@imakeflushedpage`, it is turned *true* iff $\kappa_c(\rho_t) = \infty$ and $V'_P = \pi^h(p_t)$ to mean the floats in $\kappa_c(\lambda_t)$ should be put in a float column in the last page as usual.
- In `\pcol@iflushfloats`, it is turned *true* iff one or more columns have non-empty $\kappa_c(\lambda_d)$ after shipping a page for float columns out, so that `\if@fcolmade` is let have its value after scanning all columns.
- In `\pcol@output@end`, it is turned *true* iff we built float columns, or the main vertical list in the last page is empty and the page is not the starting page, so that we create a new page for the post-environment stuff.
- In `\globalcounter{ctr}`, it is turned *true* iff $\langle ctr \rangle \in \Theta^g$ already.
- In `\pcol@cmptrrelt(\theta)`, it is turned *true* iff θ is not in Θ_0 or $val(\theta) \neq val_0(\theta)$, so that θ is added to `\@gtempa` being the list of local counters to be synchronized.
- In `\pcol@switchcolumn[d]`, it is turned *false* iff $0 \leq d < C$ so that we complain c is invalid if the switch is *true*.
- In `\pcol@ac@caption@def\langle s \rangle \langle t \rangle`, `\@tempwattrue` or `\@tempwafalse` is given as its first argument s by `\pcol@ac@caption@enable` or `\pcol@ac@caption@disable` respectively, so that `\if@ac@caption@if@t` is made `\let`-equal to s and `\pcol@ac@caption` examines it for enabling/disabling `\addcontentsline` respectively. The macros `\pcol@ac@caption@if@lof` and `\pcol@ac@caption@if@lot` are initialized to be `\let`-equal to `\@tempwattrue` as the default.
- In `\pcol@icolumncolor`, it is turned *true* iff we are in a `\vbox` or in restricted horizontal or math mode.
- In `\pcol@backgroundcolor@i`, it is examined if the root of the invocation chain is `\backgroundcolor` which turns the switch *true*, or `\nobackgroundcolor` which turns it *false*, to determine whether the background of a region is painted or not.

2.2 Macros

2.2.1 Procedural Macros

`\par` is T_EX's primitive to end/start paragraphs, but may be modified by L^AT_EX to have some special functionality occasionally. The macro `\pcol@output` makes it `\let`-equal to `\@@par` in which the T_EX's original definition is kept, while `\pcol@zparacol` and `\pcol@par` use it as is.

`\space` is an API macro to have a space token. It is used in `\pcol@output`, `\pcol@icolumncolor`, `\pcol@defcseprulecolor@i` and `\pcol@backgroundcolor@ii` for warning messages, and in `\pcol@def@extract@fil` to `\define` the macro `\pcol@extract@fil` having spaces in its argument specification.

`\nointerlineskip` is an API macro to let `\prevdepth = -1000pt` to inhibit T_EX's baseline progress mechanism. It is used in `\pcol@ioutputelt`, `\pcol@makeflushedpage` and `\pcol@imakeflushedpage` to joint boxes without `\baselineskip` between them, in `\pcol@outputpage@ev` to suppress the `\baselineskip` insertion after the first box of painted background in the final ship-out image, and in `\pcol@bg@paint@i` for the same purpose for the box having painted backgrounds.

`\offinterlineskip` is an API macro to let `\baselineskip = -1000pt`, `\lineskip = 0` and `\lineskiplimit = \maxdimen` to suppress `\baselineskip` insertion for all boxes following this macro. It is used in `\pcol@bg@paint@i` to do that in the box in which painted backgrounds are built.

`\thepage` is an API macro to have the representation of the counter `page`. It is used in `\pcol@output` for a warning message.

`\the θ` is an API macro to have the representation of the counter θ . The macro is kept in `\pcol@thectr θ` by `\pcol@thectrelt` which also makes the macro `\let`-equal to `\pcol@thectr θ .0` if the local representation of θ in the column 0 is defined by `\definethecounter`. The macro `\pcol@setctrelt` also makes this overriding for the column c to which the macro belongs if `\pcol@thectr θ .c` being the local representation for c is defined, while it makes `\the θ` `\let`-equal to `\pcol@thectr θ` otherwise to give it its original definition.

`\stepcounter $\langle\theta\rangle$` is an API macro to increment the counter θ and zero-clear its descendant counters. It is used in `\pcol@startpage` for the counter `page`, and in `\pcol@iifootnotetext` for footnote.

`\nobreak` is an API macro to insert `\penalty = 10000` to inhibit line or page breaks. It is used in `\pcol@output@start`, `\pcol@restartcolumn`, `\pcol@icolumncolor`, `\pcol@set@color@push`, `\pcol@reset@color@pop` and `\pcol@reset@color@mpop` to meet the page-break inhibition request made by `\if@nobreak = true`.

`\addvspace $\langle skip \rangle$` is an API macro to insert a vertical $\langle skip \rangle$ if `\lastskip < $\langle skip \rangle$` , or a skip of $\langle skip \rangle + \text{\lastskip}$ otherwise. The macro is used in `\pcol@zparacol` when it finds the `paracol` environment to start is at the very beginning of a `list`-like environment, to insert `\@topsep` instead of `\parskip + \itemsep` going to be inserted by the first `\item`.

`\addpenalty $\langle pen \rangle$` is an API macro to insert a page break `\penalty = $\langle pen \rangle$` if `\if@nobreak = false`. The `\penalty` is inserted removing the last vertical skip which is reinserted after the `\penalty`. The macro is used in `\pcol@output@start` and `\pcol@restartcolumn` to insert `\interlinepenalty` if `\if@nobreak = false`, while `\pcol@zparacol` uses it to insert `\@beginparpenalty` when it finds the `paracol` environment to start is at the very beginning of a `list`-like environment.

`\footnotesize` is an API macro to set the font size for footnotes. It is used in `\pcol@fntextbody` for footnote typesetting.

`\rule $\langle r \rangle$ $\langle w \rangle$ $\langle h \rangle$` is an API macro to draw a vertical rule of w width and h tall, optionally raised by r . It is used in `\pcol@fntextbody` to have the rule of $w = r = 0$ and $h = \text{\footnotesep}$ to make the first line of the footnote is at least as tall as `\footnotesep`.

`\addcontentsline $\langle file \rangle$ $\langle sec \rangle$ $\langle entry \rangle$` is an API macro to put `\addtocontents` for the arguments to `.aux` file. The original definition of the macro is kept in `\pcol@addcontentsline` so that `\pcol@ac@disable@toc` and `\pcol@ac@caption` make the macro regain its original definition after temporarily disabling its function by making it `\let`-equal to `\pcol@gobblethree`.

`\marginpar $\langle left \rangle$ $\langle right \rangle$` is an API macro to put marginal note $\langle left \rangle$ or $\langle right \rangle$ to the left or right margin. In `\pcol@zparacol` it is made `\let`-equal to `\pcol@marginpar` for the emulation of `\marginnote`, while its original version is kept in `\pcol@mmarginpar`.

`\footnote[num]{text}` is an API macro to give a footnote $\langle text \rangle$ optionally with its number $\langle num \rangle$. In `\pcol@zparacol` it is made `\let`-equal to `\pcol@footnote` to implement its starred version and the adjustment of footnote at `\end{paracol}`, while its original version is kept in `\pcol@@footnote`.

`\footnotemark[num]` is an API macro to give a footnote mark optionally with the number $\langle num \rangle$ which the mark represents. In `\pcol@zparacol` it is made `\let`-equal to `\pcol@footnotemark` to implement its starred version and the adjustment of footnote at `\end{paracol}`, while its original version is kept in `\pcol@@footnotemark`.

`\footnotetext[num]{text}` is an API macro to give a footnote $\langle text \rangle$ optionally with its number $\langle num \rangle$ but without putting the mark in the footnoted text. In `\pcol@zparacol` it is made `\let`-equal to `\pcol@footnotetext` to implement its starred version, while its original version is kept in `\pcol@@footnotetext`.

`\footnoterule` is an API macro to draw a horizontal line above footnotes, or to insert whatever it has above them. With page-wise footnote typesetting, it is redefined in `\pcol@zparacol` so that it refers to `\textwidth` instead of `\columnwidth` for drawing the horizontal line or whatever defined, while the original version is kept in `\pcol@footnoterule`. Then it is used in `\pcol@putfootins` to separate footnotes from the stuff above them, with the original or modified definition.

`\newpage` is an API macro to break a page. It is used in `\pcol@switchcol` as the argument of `\pcol@visitallcols` to break the column-pages visited in the column-scan when the synchronized column-switching requires explicit page breaks.

`\dblfigrule` is an API macro to draw a horizontal line between the last page-wise floats and the main vertical list, or to insert whatever it has between them. The macro is used in `\pcol@startpage` to build spanning stuff in the page p in $\pi^b(p)$.

`\topfigrule` is an API macro to draw a horizontal line between the last column-wise top float and the main vertical list, or to insert whatever it has between them. The macro is used in `\pcol@cflt` and `\pcol@synccolumn` to insert it below the last (real) top float. It is also made `\let`-equal to `\relax` temporarily by `\pcol@imakeflushedpage` when it put floats in a float column as top floats. Note that the macro and its bottom counterpart `\botfigrule` should produce a vertical list whose total height and depth is 0, because L^AT_EX's float mechanism and thus our macros believe so.

`\normalcolor` is an API macro to have color specification stuff for normal coloring. The macro is used in `\pcol@putfootins` to specify the color of footnotes to be put in `\@outputbox`, in `\normalcolumncolor[c]` to define that the default color of the column c is the normal color, in `\normalcolseprulecolor` to specify that the color for column-separating rules is `\normalcolor`, and in the initial definition of `\pcol@colseprulecolor` to give the default color for column-separating rules.

`\color[mode]{color}` is an API macro defined in coloring packages to start text coloring with $\langle color \rangle$ optionally with $\langle mode \rangle$. The macro is used in `\pcol@xcolumncolor[mode]{color}[c]` and `\pcol@ycolumncolor{color}[c]` to define the default color of the column c is $\langle color \rangle$ optionally with $\langle mode \rangle$, in `\pcol@defcseprulecolor@x` and `\pcol@defcseprulecolor@y` to define the color of column-separating rules, and in `\pcol@backgroundcolor@x` to define the color for background painting of a region.

`\pfmtname` is an API macro defined in pL^AT_EX to have its format name pL^AT_EX2_ε (so far). It is used in the top level assignment of the constant switch `\ifpcol@bfbottom`.

`\PackageError⟨pkg⟩⟨msg⟩⟨hlp⟩` is a developer's API macro to stop the execution showing the error message `⟨msg⟩` with the package identification `⟨pkg⟩` and the help message `⟨hlp⟩`. The macro is used in the following macros; `\pcol@ovf` on `\@freelist` shortage; `\pcol@set@color@push` on too many math-mode colorings; in `\pcol@zparacol` on two-column typesetting outside `paracol` and illegal nesting of `paracol`; `\pcol@setcw@calcf⟨x⟩⟨y⟩⟨z⟩` on too large x/y ; `\pcol@switchcolumn` on invalid target column; `\pcol@switchenv` on illegal column-switching commands/environments in a column-switching environment; `\addcontentonly` on unknown contents type; `\footnoteplacement` on unknown layout; `\pcol@twosided` on unknown two-sided typesetting feature; `\pcol@backgroundcolor` on unknown region of background painting; and in `\pcol@backgroundcolor@i` on a region not being a column or column-separating gap but its ordinal being specified.

`\PackageWarning⟨pkg⟩⟨msg⟩` is a developer's API macro to report a warning message `⟨msg⟩` with the package identification `⟨pkg⟩`. The macro is used in `\pcol@ignore` to complain an API macro appears in `paracol` inappropriately, in `\pcol@fntextbody` if the footnote is taller than `\textheight - \skip\footins`, in `\pcol@mn@warning` to show `\marginnote` is emulated, and in `\pcol@icolumncolor`, `\pcol@defcseprulecolor@i` and `\pcol@backgroundcolor@ii` to complain `\columncolor/\normalcolumncolor`, `\colseprulecolor/\normalcolseprulecolor` or `\backgroundcolor` is used without coloring packages respectively.

`\PackageInfo⟨pkg⟩⟨msg⟩` is a developer's API macro to report an informational message `⟨msg⟩` with the package identification `⟨pkg⟩`. The macro is used if the `footmisc` package is loaded to inform about the `\footnotelayout` alias being unavailable.

`\@@par` is an internal macro to keep T_EX's original primitive `\par` in it. The macro is used in `\pcol@output` to let `\par` act with its original definition, and in `\pcol@switchcol` and `\pcol@flushclear` as the argument given to `\pcol@visitallcols` to give T_EX's page builder the chance of page break in column-scanning.

`\@height` is an internal macro having the keyword `height`. It is used in `\pcol@buildcolseprule`, `\pcol@buildcselt`, `\pcol@bg@paintregion@i`, `\pcol@output@start`, and `\pcol@putbackmvl` to draw a `\hrule` for column-separating rule in the first two, a `\vrule` to be painted in the third, and an invisible `\hrule` in the fourth and last.

`\@width` is an internal macro having the keyword `width`. It is used in `\pcol@buildcolseprule`, `\pcol@buildcselt`, `\pcol@bg@paintregion@i`, `\pcol@output@start`, and `\pcol@putbackmvl` to draw a `\hrule` for column-separating rule in the first two, a `\vrule` to be painted in the third, and an invisible `\hrule` in the fourth and last.

`\@plus` is an internal macro having the keyword `plus`. It is used in the following macros.

- `\pcol@makecol` to `\define \@textbottom` with the body of a vertical skip with small infinite stretch and shrink.
- `\pcol@combinefloats` for a skip of the same amount in `\@textbottom` above and that of negative amount to *move* the effect.
- `\pcol@hfil` for skips having `1 fil` infinite stretch with g_c or $g_c/2$ to make it sure the series of columns and column-separating gaps does not cause underfull.

- `\pcol@synccolumn` to put a `1fil` infinite stretch below the main vertical list together with a small infinite shrink in the column-page being flushed and having a synchronization point, and a vertical skip with a small infinite stretch to push up the main vertical list above a synchronization point.
- `\pcol@setcw@getspec@i` to add `0pt plus 1000pt minus 1000pt` to `\@tempskipa` to ensure the register have stretch and shrink components.
- `\pcol@setcw@fill` to let `\@tempskipa = 0pt plus ffil` as the infinite stretch factor of f .

It is also used in the top level assignment of `0pt plus 1fil minus 1fil` to `\@tempskipa` for the invocation of `\pcol@defkw`.

`\@minus` is an internal macro having the keyword `minus`. It is used in the following macros.

- `\pcol@makecol` to `\define \@textbottom` with the body of a vertical skip with small infinite stretch and shrink.
- `\pcol@combinefloats` for a skip of the same amount and that of negative amount.
- `\pcol@synccolumn` to put a small infinite shrink together with a stretch of `1fil` at the bottom of the main vertical list in a column-page being flushed and having a synchronization point.
- `\pcol@setcw@getspec@i` to add `0pt plus 1000pt minus 1000pt` to `\@tempskipa` to ensure the register have stretch and shrink components.

It is also used in the top level assignment of `0pt plus 1fil minus 1fil` to `\@tempskipa` for the invocation of `\pcol@defkw`.

`\hb@xt@` is an internal macro having the sequence “`\hbox to`”. It is used in `\pcol@ioutputelt`, `\pcol@imakeflushedpage` and `\pcol@iflushfloats` to put each column-page in a `\hbox` of `\columnwidth` wide and to enclose all of them in a `\hbox` of `\textwidth` wide.

`\@namedef<cs><body>` is an internal macro to do `\def\<cs>\{body}`. It is used in the following macros.

- `\pcol@zparacol` for `\column*` and `\pcol@com@column*`.
- `\pcol@remctrelt<theta>` for `\cl@theta`.
- `\definethecounter<theta><c><rep>` for `\pcol@thectr@theta.c`,
- `\pcol@loadctrelt<theta><val_c(theta)>` for `\pcol@ctr@theta`.
- `\pcol@defcolumn` for `\pcol@com@column*`.
- `\pcol@defcseprulecolor@i` for `\pcol@colseprulecolor[.c]`.

We also use this macro in top level `\definitions` of `\pcol@com@nthcolumn*`, `\pcol@com@leftcolumn*` and `\pcol@com@rightcolumn*` for the starter of the environments `nthcolumn*`, `leftcolumn*` and `rightcolumn*`.

`\@nameuse<cs>` is an internal macro to do `\<cs>`. It is used in the following macros.

- `\pcol@bg@addext<z>\{s\}{d}` for `\pcol@bg@ext@.d.@{a.@.c,a}`.
- `\pcol@bg@columnleft` for `\pcol@columnwidth.c` and `\pcol@columnsep.c`.

- `\pcol@bg@columnwidth` for `\pcol@columnwidth·c`.
- `\pcol@bg@columnsep` for `\pcol@columnsep·c`.
- `\pcol@ccuse` for $\gamma_0^c = \text{\pcol@columncolor@box·c}$ or $\hat{\gamma}_0^c = \text{\pcol@columncolor·c}$.
- `\column*` for `\pcol@com@column*`.
- `\pcol@zparacol` for `\pcol@colpream·0`.
- `\pcol@storetrelt(θ)` for `\pcol@ctr@ θ` .
- `\pcol@cmpctrelt(θ)` for `\c@ θ` and `\pcol@ctr@ θ` .
- `\pcol@synccounter` for `\pcol@counters·c` for the column c .
- `\pcol@syncctrelt(θ)` for `\c@ θ` .
- `\pcol@stepcounter(θ)` for `\pcol@counters·c` for the column c , and for `\c1@ θ` .
- `\pcol@switchcol` to the column c for `\pcol@colpream·c`.
- `\pcol@aonlyelt(t) $\langle c \rangle$` for `\pcol@ac@def@ t` .
- `\pcol@ac@def@lof($eord$)` and `\pcol@ac@def@lot($eord$)` for `\pcol@ac@caption@ $\langle eord \rangle$` .
- `\pcol@ac@caption($type$) $[\langle lcap \rangle]$ $\langle cap \rangle$` for `\pcol@ac@caption@if@ t` and for `\ext@ $\langle type \rangle$` .
- `\footnoteplacement{ l }` for `\pcol@fnlayout@ l` .
- `\pcol@twosided[T]` for `\pcol@twosided@ t` where $t \in T$.

`\@gobble $\langle arg \rangle$` discards its argument $\langle arg \rangle$. It is used in `\pcol@output@start`, `\pcol@icolumncolor` and `\pcol@set@color@push` for temporarily letting `\aftergroup` be `\@gobble` to nullify `\aftergroup` with `\reset@color` invoked in `\pcol@set@color`, being the original version of `\set@color`, and in `\pcol@zparacol` to make `\pcol@bg@paintbox` `\let-equal` to `\@gobble` to nullify it if any coloring packages have not been loaded. In addition, the macros `\pcol@F` and `\pcol@Fe` for logging are made `\let-equal` to `\@gobble` at the top level to nullify them.

`\@ifundefined $\langle cs \rangle$ $\langle then \rangle$ $\langle else \rangle$` is an internal macro to do $\langle then \rangle$ or $\langle else \rangle$ if $\langle cs \rangle$ is undefined or defined respectively. It is used in the following macros; `\pcol@bg@paintregion(a) $\langle c \rangle$` to examine if either `\pcol@bg@color@ a · a · c` or `\pcol@bg@color@ a` is defined; `\addcontentsonly $\langle t \rangle$ $\langle c \rangle$` to stop the execution if `\pcol@ac@def@ t` is not defined; `\footnotelayout{ l }` to stop the execution if `\pcol@fnlayout@ l` is not defined; `\pcol@twosided[T]` to stop the execution if T has t such that `\pcol@twosided@ t` is not defined; `\pcol@backgroundcolor` to stop the execution if `\pcol@bg@@ a` is not defined for a region a ; and `\pcol@backgroundcolor@i` to stop the execution if `\pcol@bg@mayhavecol· a` is not defined for a region a .

`\@ifnextchar $\langle char \rangle$ $\langle then \rangle$ $\langle else \rangle$` is an internal macro to do $\langle then \rangle$ or $\langle else \rangle$ if the character following to the macro is $\langle char \rangle$ or not respectively. It is used in the following macros to examine if they are followed by a '['.

```

\paracol, \pcol@zparacol, \columnratio,
\pcol@com@column* (initial definition), \pcol@com@switchcolumn,
\pcol@iswitchcolumn, \pcol@adjustfnctr, \pcol@iffootnotetext,
\twosided, \marginparthreshold, \columncolor, \pcol@columncolor,
\normalcolumncolor, \colseprulecolor, \pcol@defcseprulecolor,
\normalcolseprulecolor, \pcol@backgroundcolor,
\pcol@backgroundcolor@w.

```

It is also used in `\pcol@backgroundcolor@iii` and `\pcol@backgroundcolor@iv` if they are followed by a ‘*C*’.

`\@ifstar<then><else>` is an internal macro to do *<then>* or *<else>* if the character following to the macro is ‘*’. It is used in `\pcol@yparacol`, `\globalcounter`, `\pcol@switchcolumn`, `\pcol@footnote`, `\pcol@footnotemark` and `\pcol@footnotetext` to examine if the optional ‘*’ is specified.

`\@whilesw<sw>\fi<body>` is an internal macro to iterate *<body>* while the switch *<sw>* is *true*. It is used in `\pcol@output`, `\pcol@startpage`, `\pcol@output@clear`, `\pcol@flushfloats`, `\pcol@freshpage` and `\pcol@output@end` to iterate building process of float columns or float pages while `\if@colmade = true`, and in `\pcol@switchcol` and `\pcol@flushclear` to iterate the height check of synchronized or flushed pages while `\ifpcol@flush = true`.

`\@whilenum<ifnum>\do<body>` is an internal macro to iterate *<body>* while the integer comparison expression *<ifnum>* is *true*. The macro is used in the following macros to iterate their own procedures for all columns $c \in [0, C)$.

```
\pcol@checkshipped, \pcol@output@start, \pcol@output@switch,
\pcol@sync, \pcol@makeflushedpage, \pcol@freshpage,
\pcol@output@end, \pcol@synccounter, \pcol@com@syncallcounters,
\pcol@stepcounter, \pcol@visitallcols.
```

The macro is also used in the following macros for the ranges following macro name, where $(C^0, C^1) \in \{(0, C_L), (C_L, C)\}$ and c is the column they are working on.

```
\pcol@ioutputelt [C0, C1)
\pcol@bg@paint@ii [Cb0, C1-1)
\pcol@bg@columnleft [Cb0, c)
\pcol@addmarginpar [C0, c') (c' ∈ {c, C1 - (c - C0) - 1})
\pcol@imakeflushedpage [C0, C1)
\pcol@iflushfloats [C0, C1)
\pcol@setcw@scan [C0, C1)
```

The other users have a little bit more complicated range as follows.

- `\pcol@flushcolumn` to iterate float column building for a column c in pages q such that $q \in (\kappa_c(\beta^p), p_t)$.
- `\pcol@setcolwidth@r` to make assignment of w_c for $c \in [\min(C^0+k, C^1-1), C^1)$ where k is the number of fractions given as the first or second argument of `\columnratio` and kept in `\pcol@columnratioleft` or `\pcol@columnratoright`, respectively.
- `\pcol@setcw@calcf<x><y><z>` to calculate $\lceil y/2^{k_2+k_3} \rceil$ finding k_3 by iterating $y/2$ until the result becomes less than 2^{15} , to calculate $z'/2^k$ with the range $[0, k)$, to calculate $z'/2^{k-16}$ with the range $[0, k-16)$, and to calculate $z' \cdot 2^{16-k}$ with the range $[0, 16-k)$, where $z' \cdot 2^{16-k} = Z = z \times 1 \text{ pt}$ and k_2, k_3 and k are scaling parameters for good approximation.

`\@whiledim<ifdim>\do<body>` is an internal macro to iterate *<body>* while the dimensional comparison expression *<ifdim>* is *true*. The macro is used in `\pcol@setcw@calcf<x><y><z>` twice, at first to find $k_1 = \min\{k \mid x \cdot 2^k \geq 2^{13} \text{ pt}\}$ and to have $x \cdot 2^{k_1}$, and then to find $k_2 = \max\{k \mid y \bmod 2^k = 0\}$ and to have $y/2^{k_2}$.

`\@for<cs>:=<list>\do<body>` is an internal macro to iterate $\langle body \rangle$ for each element of the comma-separated $\langle list \rangle$ letting $\langle cs \rangle$ have the element. The macro is used in `\pcol@setcolwidth@r` to scan its argument $\langle ratio \rangle$ defined by `\columnratio`, and in `\pcol@setcw@scan` to scan its argument $\langle spec \rangle$ defined by `\setcolumnwidth`.

`\@tfor<cs>:=<list>\do<body>` is an internal macro to iterate $\langle body \rangle$ for each non-space token in $\langle list \rangle$ letting $\langle cs \rangle$ have the token. The macro is used in `\pcol@bg@paint@ii`, `\pcol@setcw@getspec` and `\pcol@twosided` to scan their arguments, in `\pcol@setcw@getspec@i` to scan a column/gap specification to remove spaces from it, and in `\pcol@twosided[T]` to scan all tokens being two-sided typesetting features in T .

`\@next<elm>\<lst>\<suc>\<fail>` is an internal macro to remove the first element from $\langle lst \rangle$, `\define` $\langle elm \rangle$ to have the first element, and then do $\langle suc \rangle$, if $\langle lst \rangle$ is not empty. Otherwise, it performs $\langle fail \rangle$. The macro is used in the following macros to obtain an `\insert` from `\@freelist`.

- `\pcol@opcol` for the completed column-page.
- `\pcol@startpage` for float pages and spanning stuff for page-wise top floats.
- `\pcol@output@start` for the pre-environment stuff, and column-pages and γ_0^c of all columns.
- `\pcol@output@switch` for the column-page from which we are leaving.
- `\pcol@iscancst` for γ_0^c .
- `\pcol@savefootins` for footnotes.
- `\pcol@flushcolumn` for float columns and the empty column-page in p_t .
- `\pcol@synccolumn` for an MVL-float on a synchronization if its point defined by a column whose main vertical list is empty.
- `\pcol@output@end` for the page-wise floats in the last page if the main vertical list of the page is empty.
- `\pcol@icolumncolor` for γ_0^c .

The macro is also used in `\pcol@ioutputelt` to obtain completed column-pages from S_c .

`\@xnnext\@elt<car>\<cdr>\@<first>\<rest>` is an internal macro to remove the first element `\@elt<car>` from a list in the form of `\@elt e_1 \dots \@elt e_n` where $\langle cdr \rangle = \@elt e_2 \dots \@elt e_n$ and `\define` $\langle first \rangle$ as $\langle car \rangle$ and globally `\define` $\langle rest \rangle$ as $\langle cdr \rangle$. It is used in `\pcol@addmarginpar` to get the first element of `\@currlist` being a `\insert` for a right marginal note without modifying `\@currlist`.

`\@cons<lst>\<elm>` is an internal macro to add `\@elt<elm>` to the tail of $\langle lst \rangle$.

- `\pcol@makecol` to add $span(H, h)$ to the tail of $\pi^s(p_t)$.
- `\pcol@opcol` to add the completed current column-page $\kappa_c(\beta)$ to S_c .
- `\pcol@startpage` to add $\pi(p_t - 1)$ and float pages to Π .
- `\pcol@outputelt` to return spanning stuff $\pi^i(q)$ in a shipped-out float page q to `\@freelist`, or to add $\pi(q)$ to Π if the page q is kept.
- `\pcol@ioutputelt` to return spanning stuff $\pi^i(q)$, page-wise footnotes $\pi^f(q)$ and/or column-pages $s_c(q)$ for all $c \in [0, C]$ in a shipped-out page q to `\@freelist`.
- `\pcol@output@start` to return the current column-page $\kappa_0(\beta)$ to `\@freelist`.

- `\pcol@output@switch` to add $\text{span}(H, h)$ to the tail of $\pi^s(p_t)$.
- `\pcol@restartcolumn` to return the current column-page $\kappa_c(\beta)$ to be resumed, and its footnotes $\kappa_c(\tau)$ if any, to `\@freelist`.
- `\pcol@getmparbottom<t><h>` to add $\text{mpar}(\max(t, b_n), \max(t, b_n)+h)$ to the tail of the list $M_{\{L,R\}}^{\{l,r\}}$, and its callee `\pcol@getmpbelt<t_i><b_i>` to add $\text{mpar}(t_i, b_i)$ or $\text{mpar}(\max(t, b_{i-1}), \max(t, b_{i-1})+h)$ to the so-far tail of the list in rebuilding.
- `\pcol@flushcolumn` to return footnotes $\kappa_c(\tau)$ in the current column-page of c to be flushed to `\@freelist` if any, and to add the flushed column-page and float columns to S_c .
- `\pcol@makefcolelt` to add a float to `\@toplist` or $\kappa_c(\lambda_d)$.
- `\pcol@synccolumn` to add an MVL-float for synchronization to $\kappa_c(\lambda_t)$.
- `\pcol@makeflushedpage` to return spanning stuff $\pi^i(p_t)$ and/or page-wise footnotes $\pi^f(p_t)$ in the top page to be flushed to `\@freelist` if any.
- `\pcol@imakeflushedpage` to return column-wise footnotes in $\kappa_c(\tau)$ s.t. $\kappa_c(\beta^p) = p_t$ to `\@freelist` if any.
- `\pcol@output@end` to return $\pi^f(p_t)$, all current column-pages $\kappa_c(\beta)$, and all $\gamma_0^c \neq \perp$ to `\@freelist`.
- `\pcol@end@dblfloat` to add a page-wise float in `\@currbox` to `\@dbldeferlist`.
- `\globalcounter<\theta>` to add a global counter θ to Θ^g .
- `\pcol@iremctrelt<\theta>` to add a local counter θ to Θ^l .
- `\pcol@storectrelt<\theta>` to add a pair $\langle \theta, \text{val}_c(\theta) \rangle$ to Θ_c for a column c .
- `\pcol@savectrelt<\theta>` to add a pair $\langle \theta, \text{val}(\theta) \rangle$ to Θ_c for a column c .
- `\pcol@cmpctrelt<\theta>` to add a counter θ to the list of local counters to be synchronized.
- `\addcontentonly<t><c>` to add a pair $\langle t, c \rangle$ to T .
- `\pcol@backgroundcolor@ii` to add a region whose background is painted to `\pcol@bg@defined`.

`\@cdr<a_1><a_2>\cdots<a_n>\@nil` is an internal macro to be expanded to $\langle a_2 \rangle \cdots \langle a_n \rangle$. It is used in `\pcol@getcurrpinfo<cs>` to extract $\pi(p_t)$ from `\pcol@currpage = \@elt<\pi(p_t)>` and to `\define <cs>` letting it have $\pi(p_t)$.

`\protected@edef<macro><body>` is an internal macro to do `\edef<macro><body>` with the `\protection` so that `\protect<cs>` is kept in the expansion. It is used in `\pcol@fntextbody` to `\edefine \@currentlabel`.

`\@latex@warning@no@line<msg>` is an internal macro to report a warning message $\langle msg \rangle$ without the line number in which the cause lies. It is used in `\pcol@output` if a page with floats and very short main vertical list is built.

`\@eha` is an internal macro having a help message saying the command causing an error is ignored. It is used in `\pcol@zparacol`, `\pcol@setcw@calcf`, `\pcol@switchcolumn`, `\pcol@switchenv`, and `\addcontentonly` as the argument of `\PackageError`.

`\@ehb` is an internal macro having a help message saying the error causes a serious problem. It is used in `\pcol@ovf`, `\pcol@zparacol` and `\pcol@set@color@push` as the argument of `\PackageError`.

`\@parmoderr` is an internal macro to complain about misplacement of a macro or environment which is expected to appear in “outer par mode”. It is used in `\pcol@zparacol` if it finds `\ifinner = true`.

`\@Esphack` is an internal macro to put back the horizontal skip and space factor saved by `\bsphack` at the end of an environment. It is used in `\pcol@end@dblfloat`.

`\reset@font` is an internal macro `\let`-equal to `\normalfont` to use a standard font. It is used in `\pcol@fntextbody` for footnote typesetting.

`\set@color` is an internal macro to start coloring of texts following it. By default it is `\relax` but may have a definition to put a `\special` for coloring with the color in `\current@color`. In the following macros, it is examined if `\set@color = \relax` and/or some local definition is given to `\set@color`.

- `\pcol@output` lets `\set@color = \pcol@set@color` i.e., lets it regain its original definition because we don’t need any special operations in `\output` routine.
- `\@outputpage` performs background painting if `\set@color ≠ \relax`.
- `\pcol@zparacol` performs set-up operations for text coloring, including making `\set@color \let`-equal to `\pcol@set@color@push` saving its original definition into `\pcol@set@color`, and enabling background painting macros if `\set@color ≠ \relax`, while these background painting macros are nullified otherwise.
- `\pcol@icolumncolor` complains that no color packages have been loaded if `\set@color = \relax`, and then otherwise temporarily lets it be the original saved in `\pcol@set@color` to `\insert` a `\vbox` to update γ_0^c or to do the update immediately.
- `\pcol@iicolumncolor` temporarily lets `\set@color = \relax` so that `\color` or `\normalcolor` invoked in the macro just defines `\current@color` to be set into $\hat{\gamma}_0^c$ without doing any other coloring operations.
- `\pcol@defcseprulecolor@i` complains that no color packages have been loaded if `\set@color = \relax`, while otherwise the macro temporarily lets `\set@color = \relax` to invoke `\color` (or `\normalcolor`) to check if its arguments are properly given.
- `\pcol@backgroundcolor@ii` complains that no color packages have been loaded if `\set@color = \relax`, while otherwise its descendent `\pcol@backgroundcolor@x` temporarily lets `\set@color = \pcol@backgroundcolor@y` to `\define` `\pcol@bg@color@a[@-c]` to be `\current@color`.

`\reset@color` is an internal macro to finish text coloring started by `\set@color`. By default it is undefined but may have some definition to put a `\special` to finish coloring. It is used in `\pcol@clearcolorstack` so as to apply it to all elements in Γ^c , in `\pcol@iscancst` to put it to the main vertical list in the case that γ_0^c was \perp and then updated, in `\pcol@icolumncolor` to apply it to all elements in $\hat{\Gamma}^c$, and in `\pcol@reset@color@pop` and `\pcol@reset@color@mpop` to have an uncoloring `\special` in the `\vbox` for γ_i and $\gamma_{i,m}$.

`\color@begingroup` is an internal macro to open a group in which a color is specified. It is used in `\pcol@putfootins` to enclose footnotes with `\normalcolor`, and in `\pcol@fntextbody{text}` to enclose the coloring in the footnote $\langle text \rangle$.

`\color@endgroup` is an internal macro to close a group in which a color is specified. It is used in `\pcol@putfootins` to enclose footnotes with `\normalcolor`, and in `\pcol@fntextbody{text}` to enclose the coloring in the footnote $\langle text \rangle$.

`\@stpelt θ` is an internal macro to zero-clear the counter θ for the implementation of `\stepcounter`. It is used in `\pcol@stepcounter` to clear the descendent counters of a global counter θ^g listed in `\pcol@cl@ θ^g` .

`\@nbitem` is an internal macro to insert a vertical skip of `\@outerparskip` – `\parskip` above the first `\item` of a list-like environment when `\if@nobreak = true`. It is used by `\pcol@zparacol` when it finds the `paracol` environment to start is at the very beginning of a list-like environment and `\if@nobreak = true`.

`\@parboxrestore` is an internal macro to set up typesetting parameters for paragraphs encapsulated in a box. It is used in `\pcol@fntextbody` for paragraphs in a footnote.

`\@finalstrut $\langle box \rangle$` is an internal macro to add an invisible vertical rule whose depth is that of a $\langle box \rangle$. It is used in `\pcol@fntextbody` to make the last line of the footnotes is as deep as `\strutbox` at shallowest.

`\@sect` is an internal macro to implement sectioning commands. The original definition of the macro is kept in `\pcol@ac@enable@toc`, while `\pcol@ac@def@toc` makes it `\let`-equal to `\pcol@ac@enable@toc` or `\pcol@ac@disable@toc`, the latter of which uses the original version temporarily disabling `\addcontentsline`.

`\@svsechd` is an internal macro (locally) `\defined` in `\@sect` and `\@ssect` to keep the section header of a sectioning command such as `\paragraph` which puts the header as the leading text of the paragraph following the command rather than putting it as an individual paragraph. The macro is `\globalized` in `\pcol@sptext` so that it is properly referred to in `\everypar` for the paragraph led by the text in case that a spanning text has the sectioning command only and thus the `\definition` of the macro must survive after we close the group in which the spanning text is put.

`\@svsec` is an internal macro (locally) `\defined` in `\@sect` to keep `\thesection` etc. to be displayed as the leading part of the section header. The macro is `\globalized` in `\pcol@sptext` together with `\@svsechd` because it is in the body of `\@svsechd`.

`\@caption` is an internal macro to implement `\caption`. The original definition of the macro is kept in `\pcol@ac@caption@latex`, while `\pcol@ac@caption@def` makes it `\let`-equal to `\pcol@ac@caption` which uses the original version temporarily disabling `\addcontentsline` if necessary.

`\end@float` is an internal macro to close a column-wise float environment. It is used in `\pcol@end@dblfloat` (but never invoked because we have `\if@twocolumn` true always).

`\end@dblfloat` is an internal macro to close a page-wise float environment. It is replaced with our own `\pcol@end@dblfloat`.

`\@endfloatbox` is an internal macro to close a `\vbox` for a float. It is used in `\pcol@end@dblfloat`.

`\@largefloatcheck` is an internal macro to examine if a float is too large. It is used in `\pcol@end@dblfloat`.

`\@floatplacement` is an internal macro for `\output` routine to reinitialize column-wise float placement parameters. It is used in our own version of it, `\pcol@floatplacement`.

`\@dblfloatplacement` is an internal macro for `\output` routine to reinitialize page-wise float placement parameters. It is used in `\pcol@startpage` and `\pcol@output@clear` prior to processing page-wise floats in `\@dbldeferlist`. As discussed in item-(2) of §1.8, this macro in 2015 or later version of L^AT_EX lets `\f@depth = 1sp`.

`\@xympar` is an internal macro to perform the last operations for `\marginpar`. In `\pcol@zparacol` it is made `\let`-equal to `\pcol@xympar` for the emulation of `\marginnote`, while its original version is kept in `\pcol@@xympar`.

`\p@footnote` is an internal macro to have the prefix to `\thefootnote` in the printed reference of the counter `footnote`. It is used in `\pcol@fntextbody` to produce `\@currentlabel`.

`\@thefnmark` is an internal macro to have `\thefootnote`¹³¹. It is used in `\pcol@fntextbody` to produce `\@currentlabel`.

`\@footnotetext{text}` is an internal macro to implement `\footnote` and `\footnotetext` for `\langle text \rangle`. In `\pcol@zparacol`, it is made `\let`-equal to `\pcol@fntext`.

`\@makefntext{fn}` is an internal macro to typeset the footnote `\langle fn \rangle`. It is used in `\pcol@fntextbody{text}` to typeset the footnote `\langle text \rangle` with some other stuff.

`\@emptycol` is an internal macro for `\output` routine to put back an empty page to the main vertical list. It is used `\pcol@output` if a page with floats and very short main vertical list is built.

`\@specialoutput` is an internal macro for `\output` routine to process an `\output` request made by L^AT_EX's original `\clearpage`, `\end{float}` and `\marginpar`. It is used in `\pcol@specialoutput` to process the request for floats or marginal notes.

`\@opcol` is an internal macro for `\output` routine to output a page or to keep the first column until the second one is completed. This macro is used in `\pcol@output` to process a sneaked `\output` request from outside of `paracol`, and in `\pcol@output@end` for the case page-wise floats are left at `\end{paracol}` and they are put in float pages.

`\@makecol` is an internal macro for `\output` routine to build the ship-out image of a column in `\@outputbox` consisting of top floats, main vertical list in `\box255`, footnotes in `\footins`, and bottom floats¹³². It is used in `\pcol@output` to process a sneaked `\output` request from outside of `paracol`, in `\pcol@@makecol` for a column-page to be flushed by `\pcol@flushcolumn` and `\pcol@makeflushedpage`, in `\pcol@makecol` for an ordinary column-page, and in `\pcol@output@start` and `\pcol@makenormalcol` for pre-environment stuff.

`\@textbottom` is an internal macro for `\output` routine to be put at the bottom of `\@outputbox` in which a column-page is stored, by `\@makecol`. This macro is temporarily re`\defined` by `\pcol@makecol` for a column-page having synchronization points so that it has a vertical skip of infinite stretch and shrink to push up/down the stuff below the last synchronization point in order to adjust its top to the point. After that, its original definition kept in `\pcol@textbottom` is restored. Another modifiers of the macro are as follows; `\pcol@makenormalcol` to make the macro `\let`-equal to `\relax` temporarily to avoid the insertion of whatever the macro has in `\@makecol`; and `\pcol@makeflushedpage` to let the macro have `\vfil` temporarily so that empty columns in a last page are made *full size* without underfull.

¹³¹Or `\thempfootnote` in `minipage` environment.

¹³²In pL^AT_EX, the order of footnotes and bottom floats are reversed.

`\@outputpage` is an internal macro for `\output` routine to output a page kept in `\@outputbox` together with the header and footer. The original definition of this macro is saved in `\pcol@@outputpage` to be used in `\pcol@outputpage@l` and `\pcol@outputpage@r` being callees of our own revised version of `\@outputpage`. Therefore, any `\output` request to result in page ship-out reaches our own `\@outputpage` and then L^AT_EX's one after we perform operations for parallel-paging and background painting onto the ship-out image.

`\@combinefloats` is an internal macro for `\output` routine to combine top and bottom floats in `\@toplist` and `\@botlist` respectively with `\@outputbox` in which the main vertical list and footnotes¹³³ have been put by `\@makecol`, and to have the result in `\@outputbox` again. In `\pcol@zparacol`, it is made `\let`-equal to our own `\pcol@combinefloats` so that `\@makecol` and `\pcol@makenormalcol` uses it instead of the original one. However, if `\pcol@output` finds that the `\output` request for a page break outside `paracol` is sneaked into our own `\output` routine, it makes this macro `\let`-equal to `\pcol@@combinefloats` in which L^AT_EX's version is kept so that L^AT_EX's original `\output` routine works perfectly as original.

`\@cflt` is an internal macro for `\output` routine to put all top floats in `\@toplist` and related stuff such as the vertical skip of `\textfloatsep` into `\@outputbox` together with its old contents being the main vertical list and footnotes¹³³. It is used in `\pcol@combinefloats` if the column-page being processed does not have synchronization points.

`\@cflb` is an internal macro for `\output` routine to put all bottom floats in `\@botlist` and related stuff such as the vertical skip of `\textfloatsep` into `\@outputbox` together with its old contents being top floats, the main vertical list and footnotes¹³³. It is used in `\pcol@combinefloats`.

`\@comflelt<flt>` is an internal macro for `\output` routine to put `<flt>` being a column-wise top or bottom float to the tail of `\@tempboxa` which finally has all top/bottom floats in a column. It is used in `\pcol@cflt` to apply it to each element of `\@toplist`.

`\@comdblfelet<flt>` is an internal macro for `\output` routine to put `<flt>` being a page-wise float to the tail of `\@tempboxa` which finally has the spanning stuff for page-wise floats. It is used in `\pcol@startpage` to apply to each element of `\@dbltoplist` to have the spanning stuff.

`\@startcolumn` is an internal macro for `\output` routine which tries to build a float column for the floats in `\@deferlist` and, if the column is not built, tries to move floats to `\@toplist` and `\@botlist`. This macro is used in `\pcol@output` to process a sneaked `\output` request from outside of `paracol`, and in `\pcol@output@end` for the case page-wise floats are left at `\end{paracol}` after which they become column-wise ones.

`\@tryfcolumn<lst>` is an internal macro for `\output` routine which examines if a float column or a float page can be built with some floats in `<lst>` and, if so, builds the page in `\@outputbox` removing floats put in the page from `<lst>`. It is used in `\pcol@startpage` for page-wise floats in `\@dbldeferlist`, and in `\pcol@startcolumn` for column-wise floats in $\kappa_c(\lambda_d)$.

`\@scolelt<flt>` is an internal macro for `\output` routine which examines if a column-wise float `<flt>` can be added to `\@toplist` or `\@botlist` being the list of the floats to be put at the top or bottom of a page respectively. Then, if the examination succeeds, `<flt>` is added to `\@toplist` or `\@botlist`, while it is added to `\@deferlist` otherwise. It is used in `\pcol@trynextcolumn` to apply to each element of (the copy of) $\kappa_c(\lambda_d)$.

¹³³In pL^AT_EX, footnotes are not in `\@outputbox` because of the reversal of footnotes and bottom floats.

`\@sdblcolelt<flt>` is an internal macro for `\output` routine which examines if a page-wise float `<flt>` can be added to `\@dbltoplist` being the list of the floats to be put at the top of a page, and if so, adds `<flt>` to `\@dbltoplist`, while it is added to `\@dbldeferlist` or `\@deferlist` depending on L^AT_EX's version otherwise as discussed in item-(3) of §1.8. It is used in `\pcol@startpage` to apply to each element of (the copy of) `\@dbldeferlist`.

`\@addmarginpar` is an internal macro for `\output` routine to add a marginal note. Its original definition is kept in `\pcol@@addmarginpar` and used in `\pcol@addmarginpar` being our own `\@addmarginpar`.

`\@makefcolumn<lst>` is an internal macro for `\output` routine to build a float column with some floats at the head of a float list `<lst>` and to remove the floats from the list. It is used in `\pcol@flushcolumn` and `\pcol@iflushfloats` for $\kappa_c(\lambda_d)$ of column-wise floats, and in `\pcol@output@clear` for `\@dbldeferlist` of page-wise floats.

2.2.2 Structural Macros

`\@elt<a1>⋯<an>` is an internal control sequence to represent a list element having n sub-elements. The sequence is often made `\let`-equal to a macro which processes `<a1>⋯<an>` and is applied to all members in a list. It is also made `\let`-equal to `\relax` on a manipulation of a list, such as element addition and concatenation, by `\edef` or `\xdef`. The usages of the sequence are as follows.

- `\pcol@F@count` defines `\@elt` as a macro to increment `\@tempcnta` by one to measure the cardinality of `\@freelist`.
- `\pcol@cflt` lets `\@elt = \@comflelt` for $\kappa_c(\lambda_t)$, and then `\@elt = \relax` to concatenate `\@freelist` and $\kappa_c(\lambda_t)$.
- `\pcol@setpageno` lets `\@elt = \pcol@setpnoelt` for Π .
- `\pcol@setpnoelt` and `\pcol@setmpbelt` let `\@elt = \relax` to add $\pi(q)$ to Π . It also uses `\@elt` to define `\pcol@currpage` with `\@elt{\pip(q)}<\pii(q)><\pif(q)>\{\pis(p)\}\{\pim(p)\}`.
- `\pcol@defcurrpage` lets `\@elt = \relax` to `\xdefine` `\pcol@currpage` with `\@elt{\pip(q)}<\pii(q)><\pif(q)>\{\pis(p)\}\{\pim(p)\}`.
- `\pcol@nextpage` lets `\@elt = \pcol@nextpelt` for Π .
- `\pcol@getcurrpage` lets `\@elt = \pcol@getpelt` for Π^+ .
- `\pcol@startpage` lets `\@elt = \@sdblcolelt` for (the copy of) `\@dbldeferlist`, and `\@elt = \@comdblfilelt` for `\@dbltoplist`. It also lets `\@elt = \relax` for the concatenation of `\@dbldeferlist` and `\@deferlist`, and that of `\@freelist` and `\@dbltoplist`.
- `\pcol@outputcolumns` lets `\@elt = \pcol@outputelt` with two arguments for (the copy of) Π .
- `\pcol@ioutputelt` defines `\pcol@bg@footnoteheight` and `\pcol@bg@floatheight` with `\@elt` to let painting macros add elements in them to have the height of the background regions $R_{\{n,N\}}$ and $R_{\{f,F\}}$ to be painted.
- `\pcol@buildcolseprule` lets `\@elt = \pcol@buildcselet@S` and then `\@elt = \pcol@buildcselet` for $\pi^s(p)$, and then defines `\pcol@bg@columnheight` with `\@elt` to add H'_n and `\@maxdepth` or 0 for the background region $R_{\{c,g\}}^c(n+1)$ where $n = |\pi^s(p)|$.

- `\pcol@buildcset@S` defines `\pcol@bg@spanningtop` and `\pcol@bg@spanningheight` with `\@elt` to define the region $R_S(i)$.
- `\pcol@buildcset` defines `\pcol@bg@columnheight`, `\pcol@bg@spanningtop` and `\pcol@bg@spanningheight` with `\@elt` to define regions $R_{\{c,g\}}^c(i)$ and $R_s(i)$.
- `\pcol@bg@calculate⟨z⟩⟨z_0⟩{F}` lets `\@elt = \pcol@bg@advance` to let `\@elt⟨f⟩` in F do $z \leftarrow z + f$.
- `\pcol@bg@negative{F^-}` lets `\@elt = \pcol@bg@advance` to let `\@elt⟨f⟩` in F^- do $z \leftarrow z - f$, and then lets `\@elt = \pcol@bg@advance` to go back to addition.
- `\pcol@output@start` lets all floats f imported in `\@dbldeferlist` have depth 0 by defining `\@elt⟨f⟩` to do it, redefines `\pcol@bg@textheight` with `\@elt` having height-plus-depth of pre-environment stuff for background painting of it, and defines $\pi^m(0)$ having one element in M_L^l or M_L^r for `\@mparbottom` in pre-environment stuff.
- `\pcol@makenormalcol` lets `\@elt = \relax` to concatenate `\@freelist` and `\@midlist`.
- `\pcol@trynextcolumn` lets `\@elt = \@scolelt` for (the copy of) $\kappa_c(\lambda_d)$.
- `\pcol@setcurrcol` lets `\@elt = \relax` to define κ_c with $\kappa_c(\lambda_t) = \@toplist$, $\kappa_c(\lambda_m) = \@midlist$, $\kappa_c(\lambda_b) = \@botlist$ and $\kappa_c(\lambda_d) = \@deferlist$.
- `\pcol@scancst` and `\pcol@iscancst` let `\@elt = \relax` to define the list $M = (m \mid \gamma_{j,m}^- \in \Gamma_r, j \geq i)$ for $\gamma_{i,*}^-$, and then the latter defines `\@elt` as a macro with an argument m to examine $m \in M$ for $\gamma_{i,m}$.
- `\pcol@addmarginpar` lets `\@elt = \pcol@setmpbelt` for Π^+ .
- `\pcol@getmparbottom` lets `\@elt = \pcol@getmpbelt` for the list $M_{\{L,R\}}^{\{l,r\}}$. It also lets `\@elt = \relax` for the addition of $mpar(h_i, t_i)$ to the list by itself and `\pcol@getmpbelt`.
- `\pcol@mparbottom@zero` has `\@elt` in its body to have $mpar(0,0)$ for each $M \in \{M_X^x \mid X \in \{L, R\}, x \in \{l, r\}\}$ the macro has.
- `\pcol@bias@mpbout@i{y}\@elt{t}{b}\@nil` has `\@elt` in its argument specification, and defines `\reserved@b` with `\@elt` for $mpar(t + y, b + y)$.
- `\pcol@getmparbottom@last@i{y}mpar(t_1, b_1) \cdots mpar(t_n, b_n)\@nil` at first defines `\reserved@b` with `\@elt` for $mpar(y, y)$ and then defines `\@elt` to do it for $mpar(t_i, b_i)$ for all $i \in [1, n]$.
- `\pcol@makefcolumn` lets `\@elt = \pcol@makefcolelt` for (the copy of) $\kappa_c(\lambda_d)$ to examine if each float in it can be put in a float column to be built, and then define it to put floats in $\kappa_c(\lambda_t)$ into $\kappa_c(\beta^b)$.
- `\pcol@addflhd` lets `\@elt = \pcol@hdflelt` for its argument $\kappa_c(\lambda_t)$ or $\kappa_c(\lambda_b)$, and then `\@elt = \relax` to give the default.
- `\pcol@makeflushedpage` redefines `\pcol@bg@floatheight` with `\@elt` letting it be `\relax`.
- `\pcol@imakeflushedpage` defines `\pcol@bg@footnoteheight` with `\@elt`.
- `\pcol@output@end` uses `\@elt` in the argument specification of the definition of `\pcol@do@mpbout@elem`, lets `\@elt = \relax` to add the spanning stuff of the last page to the head of `\@dbldeferlist`, and uses it in the body of `\pcol@bg@textheight` and `\pcol@bg@footnoteheight` to be defined for background painting of page-wise footnotes.

- `\pcol@zparacol` lets `\@elt = \pcol@remctrelt` for Θ^g , `\@elt = \pcol@thectrelt` for Θ^l , `\@elt = \pcol@loadctrelt` for Θ_0 , `\@elt = \pcol@cmpctrelt` for Θ^l , `\@elt = \pcol@defcomelt` for `\pcol@localcommands`, and then `\@elt = \relax` to give the default.
- `\globalcounter{ctr}` defines `\@elt{\theta}` for Θ^g to check if there is $\theta \in \Theta^g$ such that $\theta = \langle ctr \rangle$.
- `\pcol@localcommands` has the sequence of `\@elt{com}` for all local commands `\langle com \rangle`.
- `\pcol@gcounters` has `\@elt{page}` as its initial definition.
- `\pcol@removecounter{\Theta'}{\theta}` lets `\@elt = \pcol@iremctrelt` for (the copy of) its argument Θ' to remove θ from it.
- `\pcol@sscounters{elt}` lets `\@elt = \langle elt \rangle`, where `\langle elt \rangle = \pcol@storectrelt` or `\langle elt \rangle = \pcol@savectrelt`, for Θ^l , and then `\@elt = \relax` to `\xdefine` Θ_c .
- `\pcol@com@synccounter{\theta}` gives `\@elt{\theta}` as the argument of `\pcol@synccounter`.
- `\pcol@synccounter{\lst}` lets `\@elt = \relax` to have the list `\langle lst \rangle` in `\reserved@a` by `\edef`, `\@elt = \pcol@loadctrelt` for Θ_c , and then `\@elt = \pcol@syncctrelt` for `\langle lst \rangle`.
- `\pcol@stepcounter{\theta}` lets `\@elt = \pcol@stpldelt` for Θ^l , `\@elt = \pcol@stpclelt` for $\zeta(\theta)$, and then `\@elt = \@stpelt` for $\zeta(\theta)$.
- `\pcol@switchcol` lets `\@elt = \pcol@setctrelt` for Θ_c , `\@elt = \pcol@aconlyelt` for T , and then `\@elt = \relax` to give the default.
- `\pcol@icolumncolor` defines `\@elt{\hat{\gamma}_i}` to apply `\reset@color` for rewinding and `\pcol@set@color` for reestablishing to each $\hat{\gamma}_i \in \hat{T}^c = (\hat{\gamma}_0^c, \hat{T})$ by `\pcol@scancst@shadow`, in which `\@elt` is explicitly applied to $\hat{\gamma}_0^c$ if it is defined and then implicitly done to $\hat{T} = \pcol@colorstack@shadow$.
- `\pcol@set@color@push` lets `\@elt = \relax` to push a color information into \hat{T} , with save/restore of its original value.
- `\resetbackgroundcolor` lets `\@elt = \pcol@resetbackgroundcolor` to scan `\pcol@bg@defined` containing `\@elt{a'_i}` to let `\pcol@bg@color·a'_i` for each of i .

`\@empty` is a macro always having nothing. Its major usages are to examine if a macro often having a list is empty, and to make such a macro empty. The following macros use `\@empty` to examine the emptiness of the objects in parentheses.

```

\pcol@combinefloats ( $\kappa_c(\lambda_t)$ ,  $\kappa_c(\lambda_b)$ )
\pcol@checkshipped ( $S_c$ )
\pcol@startpage ( $\pi(p_t)$ , \@dbltoplist)
\pcol@makenormalcol (\@botlist in pre-environment stuff)
\pcol@getmparbottom ( $\pi^m(p)$ )
\pcol@getmparbottom@last ( $\pi^m(p_t)$ )
\pcol@setmpbelt ( $\pi^m(p)$ )
\pcol@flushcolumn ( $\kappa_c(\lambda_d)$ )
\pcol@makefcolumn ( $\kappa_c(\lambda_t)$ ,  $\kappa_c(\lambda_d)$ )
\pcol@measurecolumn ( $\kappa_c(\lambda_b)$ ,  $\kappa_c(\lambda_d)$ )
\pcol@addflhd ( $\kappa_c(\lambda_t)$ ,  $\kappa_c(\lambda_b)$ )
\pcol@synccolumn ( $\kappa_c(\lambda_t)$ )

```



```

\pcol@makeflushedpage ( $\kappa_c(\lambda_d)$ )
\pcol@imakeflushedpage ( $\kappa_c(\lambda_d)$ )
\pcol@iflushfloats ( $\kappa_c(\lambda_d)$ )
\pcol@setcw@getspec@i<default><x'_d> ( $x'_d$ )
\pcol@setcw@fill<f>\fill ( $f$ )

```

The following macros use `\@empty` to empty the objects in parentheses.

```

\pcol@cflt ( $\kappa_c(\lambda_t)$ )
\pcol@setpageno ( $\Pi$ ,  $\pi(p_t)$ )
\pcol@startpage ( $\kappa_c(\lambda_d)$ , \@dbldeferlist, \@deferlist, \@dbltoplist)
\pcol@trynextcolumn ( $\kappa_c(\lambda_d)$ )
\pcol@output@start ( $\Pi$ ,  $\pi(p_t)$ ,  $\kappa_c(\lambda_d)$ )
\pcol@makenormalcol (\@midlist in pre-environment stuff)
\pcol@addmarginpar ( $\Pi$ ,  $\pi(p_t)$ )
\pcol@getmparbottom ( $M_{\{L,R\}}^{\{l,r\}}$ )
\pcol@makefcolumn ( $\kappa_c(\lambda_d)$ ,  $\kappa_c(\lambda_t)$ )
\pcol@makeflushedpage ( $\kappa_c(\lambda_t)$ )
\pcol@freshpage ( $\Pi$ ,  $\pi(p_t)$ )
\pcol@zparacol (\@gtempa)
\pcol@removecounter< $\theta'$ >\{ $\theta$ \} ( $\theta' \in \{\theta^g, C_L\}$ )
\pcol@sscounters ( $\theta_c$ )

```

`\@currentlabel` is an internal macro to have the reference to be associated with the `\label` following it. It is defined in `\pcol@fntextbody` to have the reference to the footnote with `\@thefnmark`.

`\ext@figure` is an internal macro having “lof” being the extension of the file for list of figures. It is used in `\pcol@ac@caption<type>[<lcap>]<cap>` to have “lof” when `<type> = figure`.

`\ext@table` is an internal macro having “lot” being the extension of the file for list of tables. It is used in `\pcol@ac@caption<type>[<lcap>]<cap>` to have “lot” when `<type> = table`.

`\@currbox` is an internal macro which conventionally has an `\insert` for floats, etc. The following macros use `\@currbox` having the objects in parentheses.

```

\pcol@opcol ( $\kappa_c(\beta)$ )
\pcol@startpage ( $\pi^i(p_t)$ )
\pcol@ioutputelt ( $s_c(q)$ )
\pcol@output@start ( $\pi^i(0)$ ,  $\kappa_c(\beta)$ ,  $\gamma_0^c$ )
\pcol@output@switch ( $\kappa_c(\beta)$ )
\pcol@restartcolumn ( $\kappa_c(\beta)$ )
\pcol@igetcurrcol ( $\kappa_c(\beta)$ )
\pcol@setcurrcol ( $\kappa_c(\beta)$ )
\pcol@putbackmvl ( $\kappa_c(\beta)$ )
\pcol@iscancst ( $\gamma_0^c$ )
\pcol@addmarginpar (left marginal note)
\pcol@flushcolumn ( $\kappa_c(\beta)$ ,  $s_c(q)$ )
\pcol@makefcolumn ( $\kappa_c(\beta)$ )
\pcol@measurecolumn ( $\kappa_c(\beta)$ )
\pcol@synccolumn ( $\kappa_c(\beta)$ )
\pcol@imakeflushedpage ( $\kappa_c(\beta)$ )

```

`\pcol@freshpage` ($\kappa_c(\beta)$)
`\pcol@output@end` (top float, $\kappa_c(\beta)$)
`\pcol@end@dblfloat` (page-wise float)
`\pcol@icolumncolor` (γ_0^c)

`\@marbox` is an internal macro which has an `\insert` for left marginal notes. It is used in `\pcol@xympar`.

`\@currlist` is an internal macro which has an list of `\inserts` for floats and marginal notes given to output. It is used in `\pcol@addmarginpar` to get the right marginal note from its head.

`\@freelist` is an internal macro having available `\inserts` for floats originally, but also column-pages, spanning stuff, footnotes and default column-color in our usage. Besides the acquisition of an `\insert` from it shown in the description of `\@next`, it is used by the following macros to return the the objects in parentheses to `\@freelist`.

`\pcol@cflt` ($\kappa_c(\lambda_t)$)
`\pcol@startpage` (`\@dbltoplist`)
`\pcol@outputelt` ($\pi^i(q)$)
`\pcol@ioutputelt` ($\pi^i(q)$, $\pi^f(q)$, $s_c(q)$)
`\pcol@startcolumn` ($\pi^f(p_t)$)
`\pcol@output@start` ($\kappa_0(\beta)$)
`\pcol@makenormalcol` (`\@midlist` in pre-environment stuff)
`\pcol@restartcolumn` ($\kappa_c(\beta)$, $\pi^f(p)$ or $\kappa_c(\tau)$)
`\pcol@savefootins` ($\pi^f(p)$ or $\kappa_c(\tau)$)
`\pcol@flushcolumn` ($\kappa_c(\tau)$)
`\pcol@makefcolpage` ($\kappa_c(\lambda_t)$)
`\pcol@makeflushedpage` ($\pi^i(p_t)$, $\pi^f(p_t)$)
`\pcol@imakeflushedpage` ($\kappa_c(\tau)$)
`\pcol@output@end` ($\pi^f(p_t)$, $\kappa_c(\beta)$, γ_0^c)

In addition `\pcol@F@count` scans its element to have its cardinality.

`\@nil` is an internal control sequence which is conventionally used to terminate a variable length argument. It is used in the following macros.

- `\pcol@getcurrpinfo` for the invocation of `\@cdr`.
- `\pcol@bias@mpbout@i{y}\@elt{t}{b}\@nil` to capture t and b following the convention in `\pcol@do@mpb@all@ii`.
- `\pcol@getmparbottom@last@i{y} mpar(t_1, b_1) \cdots mpar(t_n, b_n)\@nil` to capture $mpar(t_i, b_i)$ for all $i \in [1, n]$.
- `\pcol@do@mpb@all@i{M_L^l}\{M_L^r}\{M_R^l}\{M_R^r}` to terminate the list $M \in \{M_X^x \mid X \in \{L, R\}, x \in \{l, r\}\}$ in the invocation of `\pcol@do@mpb@all@ii`.
- `\pcol@do@mpb@all@ii{y} mpar(t_1, b_1) \cdots mpar(t_n, b_n)\@nil` to capture $mpar(t_i, b_i)$ for all $i \in [1, n]$, and then to terminate them passed to `\pcol@bias@mpbout@i` or `\pcol@getmparbottom@last`.
- `\pcol@setcw@scan` for the invocation of `\pcol@setcw@getspec` (w'_d)/(g'_d)/(*garbage*)`\@nil`.

- `\pcol@setcw@getspec@i` for the invocation of `\pcol@extract@fil⟨n⟩ plus ⟨f⟩ minus⟨garbage⟩\@nil` and thus in the `\definition` of it in `\pcol@def@extract@fil`.
- `\pcol@defkw` has `\@nil` in its argument specification to terminate the argument `0pt plus 1fil minus 1fil`.
- `\pcol@extract@fil@i`, `\pcol@extract@fil@ii` and `\pcol@extract@fil@iii` have `\@nil` in their argument specifications as the terminator, and thus `\@nil` appears in their invocations in `\pcol@extract@fil`, `\pcol@extract@fil@i` and `\pcol@extract@fil@ii` respectively, and in the `\definition` of `\pcol@extract@fil@iii` in `\pcol@def@extract@fil@iii`.
- `\pcol@iadjustfnctr` and `\pcol@iifootnotetext` to terminate their argument `[+−]disp` passed to `\pcol@calcfnctr`.
- `\backgroundcolor` and `\nobackgroundcolor` to terminate their first argument given to `\pcol@backgroundcolor` so that its descendants `\pcol@backgroundcolor@x` and `\pcol@backgroundcolor@z` finally capture everything not processed by their ancestors.

`\current@color` is an internal macro having color information to be put into `.dvi` as a part of the argument of coloring `\special`. It is referred to by `\pcol@bg@paintregion@i`, `\pcol@output@start`, `\pcol@icolumncolor`, `\pcol@iicolumncolor`, and `\pcol@set@color@push`, `\pcol@backgroundcolor@y`.

`\@dbldeferlist` is an internal macro having the list of page-wise floats whose page appearance are not yet fixed. It is scanned and then updated in `\pcol@startpage` and `\pcol@output@clear`, while `\pcol@output@start` lets it have `\@deferlist` made before `\begin{paracol}`, and `\pcol@output@end` adds page-wise floats to be put in the empty last page to it and then move the whole of the list to `\@deferlist`. As discussed in §1.8, 2015 or later version of L^AT_EX no longer uses this list, but we stick with it for page-wise floats produced in `paracol` environments and thus have its top level definition with empty body in `paracol`.

`\@dbltoplist` is an internal macro having the list of page-wise floats which `\@sdblcolelt` decided to be put in the new page. The macro `\pcol@startpage` scans it to put all floats into the page p_t as its spanning stuff $\pi^i(p_t)$, and then empties it after returning all floats to `\@freelist`.

`\@deferlist` is an internal macro having the list of column-wise floats whose page appearance are not yet fixed. It is scanned and then updated in `\pcol@startcolumn`, `\pcol@trynextcolumn`, `\pcol@flushcolumn`, `\pcol@makefcolumn` and `\pcol@iflushfloats`, while the following macros also act on it.

- `\pcol@output@start` moves it to `\@dbldeferlist` because it is created before `\begin{paracol}`.
- `\pcol@startpage` uses it as the interface with `\@addtodblcol` of 2015 or later version of L^AT_EX as discussed in item-(3) of §1.8.
- `\pcol@setcurrcol` and `\pcol@igetcurrcol` saves/restores it into/from $\kappa_c(\lambda_d)$, respectively.
- `\pcol@makefcolelt⟨flt⟩` returns `⟨flt⟩` to the list if `⟨flt⟩` cannot be put in the float column which the macro is working on.

- `\pcol@measurecolumn` examines its emptiness to let `\ifpcol@dfloats = true` iff not empty.
- `\pcol@output@end` lets it have `\@dbldeferlist` so that it processed as column-wise floats after `\end{paracol}`.

`\@toplist` is an internal macro having the list of column-wise floats which is decided to be put at the top of the current column-page by float environments or by `\pcol@trynextcolumn`. This list is scanned by `\pcol@cflt` if its invoker `\pcol@combinefloats` finds the macro is not empty. The list is also scanned by `\pcol@makecol`, `\pcol@output@switch` and `\pcol@measurecolumn` using `\pcol@addflhd` for the measurement of the combined size of top floats, while `\pcol@setcurrcol` and `\pcol@igetcurrcol` saves/restores the list into/from $\kappa_c(\lambda_t)$ respectively. In addition our macros may add an element or build the entire list in the following two cases. One case is for a synchronization for which `\pcol@synccolumn` lets the main vertical list in $\kappa_c(\beta^b)$ be a float, namely *MVL-float* to be added to this list. Another case is for a float column in the last page for which `\pcol@makefcolumn` and `\pcol@makefcolelt` build this list for deferred floats. In the latter case, the list is scanned by `\pcol@makefcolpage` invoked from `\pcol@makefcolumn` itself, `\pcol@flushcolumn` and `\pcol@makeflushedpage`.

`\@botlist` is an internal macro having the list of column-wise floats which is decided to be put at the bottom of the current column-page by float environments or by `\pcol@trynextcolumn`. The emptiness of this list examined by `\pcol@combinefloats` to invoke the scanner `\@cflb` unless empty, by `\pcol@output@start` to calculate the room for each column-page in the starting page, and by `\pcol@makenormalcol` to decide whether `\@makecol` is used or not for building pre-environment stuff. The list is also scanned by `\pcol@measurecolumn` for the measurement of the combined size of bottom floats, while `\pcol@setcurrcol` and `\pcol@igetcurrcol` saves/restores the list into/from $\kappa_c(\lambda_b)$ respectively.

`\@midlist` is an internal macro having the list of in-text floats which has already been put in the current column-page but is kept to check the ordering of the succeeding floats. The list is emptied by `\pcol@makenormalcol` after returning all elements in it to `\@freelist`, while `\pcol@setcurrcol` and `\pcol@igetcurrcol` saves/restores the list into/from $\kappa_c(\lambda_m)$ respectively.

`\f@depth` is an internal macro having `1sp` or being `\let`-equal to `\z@` to specify the float category, page-wise or column-wise respectively, which float-related macros work on. As discussed in item-(2) of §1.8, this feature introduced in 2015 version of L^AT_EX is nullified in `paracol` environments and thus the setting with `1sp` done by `\@dblfloatplacement` is overridden by `\pcol@startpage` and `\pcol@output@clear` when they invoke the macro.

`\cl@@ckpt` is an internal macro having the list of all counters defined by `\newcounter` and the counter `page`. The original usage of this macro is to log the values of all counters into `.aux` by `\include`, but we use it to obtain all counters in `\pcol@zparacol` and `\pcol@globalcounter@s`.

`\cl@θ` is an internal macro having the list $\zeta(\theta)$ of descendant counters of the counter θ whose increment by `\stepcounter` lets them 0. The macro `\pcol@remtrelet` moves it to `\pcol@cl@θ` and re\defines it to have `\pcol@stepcounter{θ}`.

`\reserved@a` is an internal macro for temporary use. Its usages are as follows.

- In `\pcol@Fe`, it is used to keep the cardinality of `\@freelist` in `\pcol@F@n` to log it by `\pcol@FF` after `\pcol@F@n` is let have another measurement result.
- In `\pcol@iLogLevel⟨l⟩⟨name⟩`, it is used to implement `\let⟨name⟩ = ⟨name⟩·l′` where $l′$ is the roman representation of the level l .
- In `\pcol@setpageno`, it has Π^+ so that we update Π and $\pi(p_t)$ scanning their original contents.
- In `\pcol@getcurrrpinfo`, it has $\pi(p_t)$ so that we give its five components to `\pcol@getpinfo` as its first five arguments.
- In `\@outputpage`, it has a sequence of background painting for both left and right parallel-pages to be shipped out outside `paracol` environment.
- In `\pcol@outputpage@ev`, it has the expansion result of `\meaning\yoko` to be compared with `\reserved@b` having `\string\yoko` for examining if `\yoko` is a primitive of underlying `TEX`.
- In `\pcol@bg@paintregion⟨a⟩⟨c⟩`, it is let have $a′ = a·c$ or $a′ = a$, and then referred to by `\pcol@bg@paintregion@i` to use `\pcol@bg@color·a′`, and by `\pcol@bg@addext⟨z⟩{s}{d}` to use `\pcol@bg@ext@·d·@·a′`.
- In `\pcol@specialoutput`, it is `\let`-equal to `\pcol@output@·f` corresponding to `\outputpenalty = \pcol@op@·f`, or `\@specialoutput`.
- In `\pcol@output@start`, it is let have a background painting macro `\pcol@bg@paintbox` and the definition of a parameter `\pcol@bg@textheight` for it.
- In `\pcol@output@switch`, it is `\let`-equal to `\@nbreaktrue` or `\@nbreakfalse` according to `\ifnbreak` in the leaving column to broadcast it to other columns.
- In `\pcol@ifempty⟨box⟩⟨then⟩⟨else⟩`, it has `⟨then⟩` or `⟨else⟩` according to the emptiness of `⟨box⟩`.
- In `\pcol@clearcolorstack`, it is `\defined` to put an uncoloring `\special` by `\reset@color` for its argument γ_i in `\pcol@iscancst`.
- In `\pcol@restorecst`, it is `\defined` to put a coloring `\special` in its argument γ_i by `\unvbox` done in `\pcol@iscancst`.
- In `\pcol@addmarginpar`, at first it is made let equal to 0 or C_L according to $c < C_L$ or not. Then it is let have Π^+ to be scanned to find $\pi^m(p)$.
- In `\pcol@getmparbottom@i`, it is let have one of $M_{\{L,R\}}^{\{l,r\}}$ according to the side margin which the marginal note to be added goes to, and then it is referred to by `\pcol@getmparbottom`.
- In `\pcol@setmpbelt@i`, it is let have what $\pi^m(p)$ should have after the update of a list of marginal notes in it, and then `\pcol@setmpbelt` updates $\pi(p)$ with the new $\pi^m(p)$ in the macro.
- In `\pcol@bias@mpbout{y}` and `\pcol@getmparbottom@last{y}`, it is let have `\pcol@bias@mpbout@i{y}` and `\pcol@getmparbottom@last@i{y}` respectively, so that they are invoked in `\pcol@do@mpb@all@ii`.
- In `\pcol@makeflushedpage`, it is let have an invocation of `\pcol@bg@paintbox` for page-wise floats in $\pi^i(p_t)$ together with the condition of the background painting.
- In `\pcol@output@end`, it is let have the invocation of `\pcol@bg@paintbox` for background painting of page-wise footnotes with the condition to do it and a `\definition` of `\pcol@bg@footnoteheight`.

- In `\pcol@defcomelt<com>`, it is used to implement `\let\<com> = \pcol@com@<com>`.
- In `\pcol@setcolumnwidth`, it is made `\let-equal` to `\pcol@setcolumnwidth@s` or `\pcol@setcolumnwidth@r` according to `\pcol@columnratioleft = \relax` or not.
- In `\pcol@setcolumnwidth@r`, it is used to have the fraction r_d being a comma-separated list element in its argument `<ratio>` defined by `\columnratio` and scanned by a `\@for` loop.
- In `\pcol@setcw@scan<C^0><C^1>\{spec\}`, at first it is let have `<spec>`, then the result of adding ‘,’ as many as $C^1 - C^0$ to the tail, and finally each element in the extended `<spec>` in a `\@for` loop.
- In `\pcol@setcw@getspec@i<default><x'_d>`, it is let have `<x'_d>` from which all space tokens are removed.
- In `\pcol@setcw@calc factors`, it is used as a waste basket to throw away $(W_T - W)/(F \times 1 \text{ pt})$ calculated by `\pcol@setcw@calcf`.
- In `\pcol@extract@fil@i<n>.<m·unit>\@nil`, it has `<n>.<m·unit>`, and is referred to by `\pcol@extract@fil@ii`.
- In `\globalcounter{ctr}`, it is used to have `<ctr>` for the `\ifx`-comparison with each $\theta \in \Theta^g$.
- In `\pcol@remctrelt<theta>`, it is used to implement `\let\pcol@cl@theta = \cl@theta`,
- In `\pcol@removecounter<theta'>\{theta\}`, it is used to have θ for the `\ifx`-comparison in `\pcol@iremctrelt`.
- In `\pcol@thectrelt<theta>`, it is used to implement `\let\pcol@thectr@theta = \thetheta`, and then is made `\let-equal` to `\pcol@thectr@theta·0`.
- In `\pcol@synccounter<lst>`, it has `<lst>`.
- In `\pcol@setctrelt<theta><val_c(theta)>`, it is made `\let-equal` to `\pcol@thectr@theta` or `\pcol@thectr@theta·c`.
- In `\pcol@switchenv`, it is used to save `\switchcolumn` which is redefined in the macro, and then to invoke `\switchcolumn` with the original definition.
- In `\pcol@fntext{text}`, it is `\let-equal` to `\pcol@fntexttother` or `\pcol@fntexttop` according to the footnote `<text>` is deferred or not.
- In `\pcol@calcfnctr<num>\@nil`, it has the first token of `<num>` for `\ifx`-comparison with ‘+’ and ‘-’.
- In `\pcol@twosided[T]`, it is let have each non-space token in T given by a `\@tfor` loop.

`\reserved@b` is an internal macro for temporary use. It is used in the following macros to keep a list shown in parentheses so that we update the list in the scan of list elements.

```

\pcol@startpage (\@dbldeferlist)
\pcol@outputcolumns (II)
\pcol@trynextcolumn ( $\kappa_c(\lambda_d)$ )
\pcol@makefcolumn ( $\kappa_c(\lambda_d)$ )
\pcol@removecounter<theta'>\{theta\} ( $\theta' \in \{\Theta^g, \theta\}$ )

```

In addition, it is used in the following macros.

- In `\pcol@outputpage@ev`, it has the expansion result of `\string\yoko` to be compared with `\reserved@a` having `\meaning\yoko` for examining if `\yoko` is a primitive of underlying T_EX.
- In `\pcol@bg@paint@ii`, it has a token in the arguments K_b , K_g and K_c of the macro scanned by `\@tfor`.
- In `\pcol@output@switch`, it is `\let`-equal to `\@afterindenttrue` or `\@afterindentfalse` according to `\if@afterindent` in the leaving column to broadcast it to other columns.
- In `\pcol@clearcolorstack`, it is `\defined` to put an uncoloring `\special` by `\reset@color` for its argument γ_0^c in `\pcol@scancst`.
- In `\pcol@restorecst`, it is `\defined` to put a coloring `\special` in its argument γ_0^c by `\unvcopy` done in `\pcol@scancst`.
- In `\pcol@scancst` and `\pcol@iscancst`, after the reference for the purposes shown in the two items above, it has $M = (m \mid \gamma_{j,m}^- \in \Gamma_r, j \geq i)$ for $\gamma_{i,*}^-$ and in the latter is scanned to find m for $\gamma_{i,m}$ in M .
- In `\pcol@addmarginpar`, it is made `let` equal to C_L or C according to $c < C_L$ or not.
- In `\pcol@bias@mpbout@i{y}\@elt{t}{b}\@nil`, it is `let` have $mpar(t + y, b + y)$, and in `\pcol@getmparbottom@last@i{y}mpar(t_1, b_1) \cdots mpar(t_n, b_n)\@nil` it is `let` have $mpar(y, y)$ or $mpar(t_n, b_n)$, so that they are `let` be a $M_{\{L,R\}}^{\{l,r\}}$ by `\pcol@do@mpb@all@ii`.
- In `\pcol@setcw@getspecc@i{default}\@x'_d`, it is `let` have each non-space token in $\langle x'_d \rangle$ to remove space tokens from it.
- In `\pcol@setcw@fill{f}\fill`, it is `let` have f .
- In `\pcol@extract@fil@ii{unit}\@nil`, it is `let` have $\langle unit \rangle$.
- In `\globalcounter{\theta}`, it has $\theta_i^g \in \Theta^g$ for `\ifx`-comparison with θ .
- In `\pcol@iremctrelt{\Theta'}{\theta}`, it has θ for `\ifx`-comparison with θ' to be removed from Θ^g or Θ .
- In `\pcol@calcfnctr{num}\@nil`, it has ‘+’ and then ‘-’ for `\ifx`-comparison with the first token of $\langle num \rangle$.
- In `\pcol@backgroundcolor@ii`, it has `\pcol@backgroundcolor@x` or `\pcol@backgroundcolor@z` according that the region of background painting and its color is defined or undefined.

`\reserved@c` is an internal macro for temporary use. It is used in `\pcol@startpage` to save `\@deferlist` in it and then to restore the list from it, and in `\pcol@iscancst` to have `\relax` or the macro itself to iterate the macro recursively.

`\reserved@d` is an internal macro for temporary use. It is used in `\pcol@iscancst` as a `\chardef` register to have 0 if m for $\gamma_{i,m}$ is not in the list $M = (n \mid \gamma_{j,n}^- \in \Gamma_r, j \geq i)$, or 1 if found.

`\@gtempa` is an internal macro used as a `\globally` modifiable scratchpad. Its usages are as follows.

- In `\pcol@ifempty{box}\then\else`, it has `\lastpenalty` in a `\vbox` whose value is examined outside the `\vbox` for the emptiness check of $\langle box \rangle$.

- In `\pcol@addmarginpar`, it is given to `\@xnext` as the target to have the second and successive elements of `\@currlist` which we cannot modify.
- In `\pcol@zparacol` and `\pcol@cmpctrelt`, it has the list of counters to be synchronized.
- In `\pcol@setcw@getspec@i<default><x'_d>`, it is made `\@empty` or `\relax` according to x'_d has `\fill` or not.
- In `\pcol@storetrelt`, `\pcol@savetrelt` and `\pcol@sscounters`, it has the new version of θ_c .

3 Register Declaration

3.1 `\count` Registers

Here we declare registers and switches. The first group is for `\count` registers.

`\pcol@currcol` The register `\pcol@currcol` has the zero-origin ordinal c of the column which we were in when `\output` is invoked. Therefore, for example, in the process of `\switchcolumn`, the register has c from which we are switching to another column. The register is initialized to be 0 by `\pcol@output@start`, and then set to `\pcol@nextcol = d` by `\pcol@restartcolumn` to switch to (or stay in) d . Note that these two assignments are `\global` while other macros may *locally* use the register to, for example, scan all columns $c \in [0, C)$. Besides two macros above, the following macros refer to the register to know which column we are in (or which column is processed by their invokers).

```
\pcol@Log@iii, \pcol@Log@ii, \pcol@FF, \pcol@makecol, \pcol@opcol,
\pcol@bg@columnleft, \pcol@bg@columnwidth, \pcol@bg@columnsep,
\pcol@output@switch, \pcol@getcurrcol, \pcol@setcurrcol,
\pcol@clearcolorstack, \pcol@restorecolorstack, \pcol@addmarginpar,
\pcol@getmparbottom@i, \pcol@setmpbelt@i, \pcol@invokeoutput134,
\thecolumn, \pcol@sscounters, \pcol@setctrelt, \pcol@com@switchcolumn,
\pcol@switchcol, \pcol@visitallcols, \pcol@aconlyelt, \pcol@flushclear,
\pcol@columncolor, \normalcolumncolor, \pcol@icolumncolor,
```

Among the macros above, `\columncolor` and `\normalcolumncolor` could refer to the register outside `paracol` environment and thus before the initialization by `\pcol@output@start`. Therefore, the register is also initialized to be 0 after its declaration to assure safe reference.

The following macros use the register for the scan of all $c \in [0, C)$ by themselves or their invokers.

```
\pcol@output@start, \pcol@output@switch, \pcol@sync, \pcol@flushcolumn,
\pcol@measurecolumn, \pcol@synccolumn, \pcol@makeflushedpage,
\pcol@flushfloats, \pcol@freshpage, \pcol@output@end,
\pcol@synccounter, \pcol@com@syncallcounters, \pcol@stepcounter.
```

The macros `\pcol@imakeflushedpage` and `\pcol@iflushfloats` also use the register for scanning but for $[C^0, C^1)$ given by their arguments.

In addition `\pcol@ccuse`, `\pcol@ifccdefined` and `\pcol@ccxdef` refer to the register to have the control sequence `\pcol@columncolor · c = $\hat{\gamma}_0^c$` or `\pcol@columncolor@box · c = γ_0^c` where c is for the current column or for all columns depending on their invokers.

¹³⁴Only for logging.

`\pcol@nextcol` The register `\pcol@nextcol` has the zero-origin ordinal d of the column to which we are switching, or in which we are staying, to be set into `\pcol@currcol` by `\pcol@restartcolumn`. The main usage of the register is to set the switching target in `\pcol@switchcolumn`, but other macros use it to specify the (temporary) target of `\pcol@switchcol`; the tallest column in `\pcol@sync`; 0 in `\pcol@zparacol`, `\pcol@sptext` and `\endparacol`; all in $[0, C)$ but $c = \pcol@currcol$ in `\pcol@visitallcols`; and c in `\pcol@flushclear` to stay in the current column c . The other user of this register is `\pcol@invokeoutput` but only for logging.

`\pcol@ncol` The register `\pcol@ncol` has the number of columns C given as the argument of `\paracol`, `\pcol@ncolleft` whose callee `\pcol@zparacol` being the sole modifier of the register `\gloally` assigns C to the register to give safe reference to `\@outputbox` invoked after a `paracol` is closed. In addition for the reference in `\@outputbox` before the first `paracol`, the register is initialized with zero after the declaration.

The following macros refer to the register to scan all columns $c \in [0, C)$.

```
\pcol@checkshipped, \pcol@output@start, \pcol@output@switch,
\pcol@sync, \pcol@makeflushedpage, \pcol@freshpage, \pcol@output@end,
\pcol@synccounter, \pcol@com@syncallcounters, \pcol@stepcounter,
\pcol@visitallcols.
```

The register `\pcol@ncolleft` has C_L being the number of columns in the left *parallel-page* if parallel-paging is in effect, or have C otherwise. Similar to C , the number C_L is given as the optional argument of `\pcol@zparacol` and is `\gloally` assigned to the register by the sole modifier `\pcol@zparacol`, unless the optional argument is not less than C which is assigned to C_L if this exception is found. The reason of the `\global` assignment and the initial zero-clearing after the declaration is same as that for C , i.e., for the reference in `\@outputpage` outside `paracol`.

The following macros examines if $C_L < C$, i.e., if parallel-paging is in effect.

```
\pcol@outputelt, \@outputpage, \pcol@output@start, \pcol@output@flush,
\pcol@output@clear, \pcol@makeflushedpage, \pcol@flushfloats,
\pcol@output@end, \pcol@zparacol,
```

In the macros listed above, `\pcol@outputelt`, `\pcol@makeflushedpage` and `\pcol@flushfloats` passes $[0, C_L)$ and $[C_L, C)$ to their respective callees `\pcol@ioutputelt`, `\pcol@imakeflushedpage` and `\pcol@iflushfloats` as their argument pair $[C^0, C^1)$ to let them work on the left and right parallel-pages respectively. The callees above also pass the pair to `\pcol@swapcolumn` to swap the scanning order of columns if column-swapping is in effect.

They also pass the pair to `\pcol@buildcolseprule` which then passes it to `\pcol@bg@paintcolumns` and `\pcol@bg@paintbox` by binding it to $[C_b^0, C_b^1) = [\pcol@bg@from, \pcol@bg@to)$ referred to by `\pcol@bg@paint@i` and its callee `\pcol@bg@paint@ii` to define the range of columns to be painted is $[C_b^0, C_b^1)$. Similar passing is done by (our own version of) `\@outputpage`, but it directly uses $[0, C_L)$ and $[C_L, C)$ as the sources and the target painting macros are `\pcol@bg@paintpage`, `\pcol@bg@@paintpage` and `\pcol@bg@paintbox`. Note that C_b^1 is initialized to be `\let-equal` to C because it may be referred to without binding¹³⁵.

The macro `\pcol@addmarginpar` also refers to C_L to know if the column c on which it is working on is in the left or right parallel-page, i.e., $c < C_L$ or not, to decide the margin to which a marginal note is put, and to pass $[0, C_L)$ or $[C_L, C)$ to `\pcol@swapcolumn` to calculate the distance to the left or right margin from the column. The examination of $c < C_L$ is also done in related macros `\pcol@getmparbottom@i` and `\pcol@setmpbelt@i`.

¹³⁵This meaningless reference has no harmful side effects.

Similar column-range specification is done for the argument pair $[C^0, C^1)$ of `\pcol@setcolumnwidth` invoked from `\pcol@zparacol`. Then the arguments are passed to the callees `\pcol@setcolwidth@r` or `\pcol@setcolwidth@s`, the latter of which also passes them to its callee `\pcol@setcw@scan`, to define the width of columns in $[C^0, C^1)$ and their separators.

The other references to C are made by `\pcol@com@switchcolumn` and `\pcol@switchcolumn` to examine $c < C$, to wraparound $C - 1$ to 0 for the former and to complain if $c \geq C$ for the latter.

`\pcol@page` The register `\pcol@page` has the zero-origin ordinal p of the page which we are in. The register is initialized to be 0 not only by `\pcol@output@start` to give the obvious starting point, but also by `\pcol@freshpage` for page flushing which clears $\Pi = \pcol@pages$ to give us another type of starting point. Then the register is incremented by `\pcol@nextpage` to advance one page, by `\pcol@nextpelt` to skip a float page, and by `\pcol@startpage` for a float page to be created. The other type of updates of the register is done by `\pcol@restartcolumn` which lets p be $\kappa_c(\beta^p)$ when we revisit the column c belonging to the page p . Note that, besides these `\global` updates, `\pcol@flushcolumn` locally updates the register to scan $\Pi = \pcol@pages$, and `\pcol@freshpage` also performs local updates but in more weird manner. Besides the updates discussed above, the macros `\pcol@Log@iii`, `\pcol@Log@ii`, `\pcol@FF`, `\pcol@makecol`, `\pcol@opcol`, `\pcol@setpageno`, `\pcol@getcurrpage`, `\pcol@startcolumn`, `\pcol@output@switch`, `\pcol@addmarginpar` and `\pcol@fntext` refer to the register to know which page they are operating on.

`\pcol@basepage` The register `\pcol@basepage` has the ordinal p_b of the base page being the oldest page not shipped out yet. The register is initialized to be 0 by `\pcol@output@start` and `\pcol@freshpage` together with `\pcol@page`, and then incremented by `\pcol@outputelt` when it ships the page p_b out. The macros `\pcol@setpageno`, `\pcol@nextpage`, `\pcol@getcurrpage` and `\pcol@addmarginpar` refer to the register in their scans of Π or Π^+ to know the zero-origin ordinal of the element for the current page p is $p - p_b$.

`\pcol@toppage` The register `\pcol@toppage` has the ordinal p_t of the top page having the most advanced column-pages, or *leading column-pages* in short. The register is initialized to be 0 by `\pcol@output@start` and `\pcol@freshpage` together with `\pcol@page`, and then let be $p = \pcol@page$ by `\pcol@startpage` to start a new page p . The macros `\pcol@makecol`, `\pcol@opcol`, `\pcol@startcolumn`, `\pcol@output@switch` and `\pcol@restartcolumn` refer to the register to examine if they are working on a column-page in the top page, while `\pcol@flushcolumn` and `\pcol@fntext` examines if the current column-page is behind the top page. The register is also referred to by `\pcol@Log@iii`, `\pcol@Log@ii` and `\pcol@FF` for logging.

`\pcol@footnotebase` The register `\pcol@footnotebase` is let have the value of `\c@footnote` at the start of a `paracol` environment by `\pcol@zparacol` to give the base value b_f for relative numbering of `footnote` done in `\pcol@calcfnctr` for the starred versions of `\footnote`, `\footnotemark` and `\footnotetext`. The register is also referred to by `\endparacol` to let `\c@footnote` have $b_f + n_f$ where $n_f = \pcol@nfootnotes$ shown below is the number of footnotes in the `paracol` environment to be closed.

`\pcol@nfootnotes` The register `\pcol@nfootnotes` is to accumulate the number of footnotes n_f in a `paracol` environment. Therefore, it is zero-cleared by `\pcol@zparacol`, then incremented by `\pcol@ifootnote` and `\pcol@ifootnotemark` for `\footnote` and `\footnotemark`, and finally referred to by `\endparacol` to let `\c@footnote` = $b_f + n_f$.

`\pcol@mcid` The register `\pcol@mcid` has the number of pushes of color stack by coloring commands in math mode between two consecutive invocations of `\output`. The register is zero-cleared by `\pcol@`

`output` because we are definitely in the main vertical mode and thus all pops corresponding to pushes in math mode must have been applied to `.tex`'s color stack. Then the register is referred to by `\pcol@set@color@push` when it is invoked in math mode, to increment it and then examine if it does not exceed the limit `\pcol@mcpushlimit` to mean the math-mode coloring still can be made. The macro then uses the value of the register as the identifier of the push operation given to `\output` through an `\insertion`.

```

1
2 %% Register Declaration
3
4 \newcount\pcol@currcol \global\pcol@currcol\z@
5 \newcount\pcol@nextcol
6 \newcount\pcol@ncol \global\pcol@ncol\z@
7 \newcount\pcol@ncolleft \global\pcol@ncolleft\z@
8 \newcount\pcol@page
9 \newcount\pcol@basepage
10 \newcount\pcol@toppage
11 \newcount\pcol@footnotebase
12 \newcount\pcol@footnotes
13 \newcount\pcol@mcid

```

3.2 Switches

The second declaration group is for switches.

`\ifpcol@output` The switch `\ifpcol@output` is *true* iff `\pcol@output@start` which turns the switch *true* has been invoked but `\pcol@output@end` which does *false* has not yet. Then the switch is examined by `\pcol@output` to detect an `\output` request sneaked from outside of the `paracol` environment. The other users `\@outputpage` and `\pcol@reset@color@pop` examine this switch to know if the macro is invoked inside or outside of `paracol` environment, while the macro `\pcol@output@start` temporarily turns the switch *false* when it ships out a page having pre-environment stuff only.

`\ifpcol@nospan` The switch `\ifpcol@nospan` is *true* iff a page p does not have spanning stuff, i.e., $\pi^i(p) = \perp$. It is set by by `\pcol@getpinfo` for the examination in `\pcol@ioutputelt`, `\pcol@makeflushedpage` and `\pcol@output@end`.

`\ifpcol@sync` The switch `\ifpcol@sync` is *true* iff `\pcol@output@switch` is invoked for synchronized column-switching by `\switchcolumn*` or its relative environment openers, or pre-flushing column height check prior to page flushing or environment closing. Therefore, the switch is `\globally` turned *true* by `\pcol@iswitchcolumn` and `\pcol@sptext` for the synchronizing column-switching but then temporarily turned *false* by `\pcol@switchcol` invoked by them for column-scanning and then turned *true* again by the macro. For pre-flushing column height check, the macro `\pcol@flushclear` turns the switch *true*. The other macro turns this switch is `\pcol@output@switch` at the end of which the switch is turned *false* to go back to the default state. The macros examining this switch are `\pcol@output@switch`, `\pcol@putbackmvl`, `\pcol@sync`, `\pcol@invokeoutput` (for logging) and `\pcol@switchcol`.

`\ifpcol@sptextstart` The switch `\ifpcol@sptextstart` is *true* iff `\pcol@output@switch` is invoked from `\pcol@sptext` prior to a spanning text. That is, the switch is `\globally`¹³⁶ let *true* and then *false* by

¹³⁶Not necessary to be `\globally` turned but we dare to do that to clearly distinguish that from the local turning in `\pcol@putbackmvl`.

`\pcol@sptext` before and after the invocation of synchronized `\pcol@switchcol` prior to the spanning text. Then the switch is examined by `\pcol@putbackmvl` after the synchronization to save the *pre-spanning-text stuff*, being all stuff in main vertical list prior to the synchronization, so that the spanning text is split from the stuff and is captured afterward by `\pcol@makecol` and/or `\pcol@output@switch`. The macro also locally turns the switch *false* if it does not follow the synchronization, i.e., its invocation is for column-scanning or is caused by pre-synchronization page break, to do the saving only when it follows the synchronization. The switch is also examined in `\pcol@output` to inhibit the warning and forced page break even when $\kappa_0(\beta^r) = \text{@colroom} < 1.5\text{baselineskip}$, because we may let it have a small value when the spanning text starts near the page bottom to capture the text portion in the page by `\pcol@makecol`. In addition, it is examined by `\pcol@switchcol` to invoke `\pcol@colpream-c`, where $c = -1$ if *true* or $c = \text{@pcol@currcol}$ otherwise.

The macro `\pcol@sptext` then *globally* turns another switch `\ifpcol@sptext` *true* before putting the spanning text into the main vertical list so that `\pcol@makecol` for the page break in the text and `\pcol@output@switch` for closing capture the text to place it appropriately especially when column-swapping is in effect. Then the switch is *globally* turned *false* by `\pcol@output@switch` to give the default state after it *broadcasts* `\if@nobreak`, `\if@afterindent` and `\everypar` to all columns.

`\ifpcol@clear` The switch `\ifpcol@clear` is *true* iff `\pcol@output@switch` is invoked for pre-flushing column height check, page flushing or environment closing. Therefore, the switch is turned *true* by `\pcol@flushclear` in the first case, and by `\pcol@makeflushedpage` in the latter two. These two macros also turned the switch *false* after the direct/indirect invocation of `\pcol@output@switch` to give the default state. The switch is examined by `\pcol@output@switch` and its descendants `\pcol@sync`, `\pcol@flushcolumn` and `\pcol@synccolumn` for synchronization, and by `\pcol@invokeoutput` for logging.

`\ifpcol@flush` The switch `\ifpcol@flush` is turned *true* by `\pcol@sync` iff it finds that the page to be synchronized or to be flushed is too tall because the sum of the total height of top floats and main text in a column and that of bottom floats and footnotes in another column is larger than $\pi^h(p_t) - v^f - V_E$, where v^f is the total height-plus-depth of the page-wise footnotes if p_t has them or 0 otherwise, and V_E is the amount given by `\ensurevspace` in synchronization or 0 in flushing. Then the switch is examined in `\pcol@sync` itself to restart the tallest column if *true*, in `\pcol@putbackmvl` to check if a spanning text is really to start, in `\pcol@switchcol` to have an explicit page break in each column if *true*, and in `\pcol@flushclear` also to have a page break if *true*. The last examiner `\pcol@flushclear` may turn the switch *true* when it is invoked from `\endparacol` if the last page leaves deferred and non-merged page-wise footnotes for which an explicit page break is also required.

`\ifpcol@outputflt` The switch `\ifpcol@outputflt` is used in `\pcol@outputelt` to know whether a float page is to be shipped out (*true*) or not. The switch is initialized to be *true* by `\pcol@outputcolumns` which invokes `\pcol@outputelt` for all $\pi(p) \in \Pi$. Then if `\pcol@outputcolumns` is invoked from `\pcol@opcol` to ship the oldest page p_b out, the switch is turned *false* when we visit the second non-float page. That is, all float pages following the first (oldest) page are shipped out but others are not. On the other hand, if `\pcol@outputcolumns` is invoked from `\pcol@sync` to ship out all pages including float pages in Π , the switch is kept *true* throughout all invocations of `\pcol@outputelt`.

`\ifpcol@lastpage` The switch `\ifpcol@lastpage` is used to know whether the following macros work on the *last* `\ifpcol@lastpagesave` page of a `paracol` environment to do special operations if so.

- `\pcol@combinefloats` adds `\textfloatsep` below bottom floats of each column if any so that the floats are well separated from the post-environment stuff.
- `\pcol@sync` examines V'_P instead of V_P for the pre-flushing column height check.
- `\pcol@makefcolumn` tries to make deferred floats as top floats.
- `\pcol@makeflushedpage` builds a short page of V'_P tall, leaves spanning stuff from ship-out if $V'_P = \pcol@colht = -\infty$ so that it becomes a float in post-environment stuff, and leaves page-wise footnotes untouched if merged footnote typesetting is in effect. This macro itself turns this switch *false* if `\ifpcol@dfloats = true` to mean one or more columns in the last page have deferred column-wise floats and thus the last page must be *full size*.
- `\pcol@imakeflushedpage` leaves the background of page-wise footnotes unpainted, and lets the depth of the last page be 0 in background painting and packing of column-pages.

The switch is initialized to be *false* by `\pcol@zparacol`, then turned *true* by `\endparacol`, and then finally turned *false* by `\pcol@output@end` again for float pages following the last page if any. The macro `\pcol@flushcolumn` saves the switch into `\ifpcol@lastpagesave` then turning it *false* during it works on column-pages in non-top and thus non-last pages to keep `\@makecol` and `\pcol@makefcolumn` from misunderstanding the pages are last, and then restore the switch when the macro reaches to the top page. This saving and temporary turning *false* is also done in `\pcol@flushclear` when it forces a page break so that the `\output` routine working on the broken non-last page correctly recognizes that. Another temporary turning is made by `\pcol@makenormalcol` but in reverse to let the switch be *true* so that its indirect callee `\pcol@combinefloats` puts a vertical skip of `\textfloatsep` below the bottom floats in pre-environment stuff.

`\ifpcol@scfnote` The switch `\ifpcol@scfnote` is turned *true* by `\pcol@fnlayout@p` and `\pcol@fnlayout@m` (through `\pcol@fnlayout@p`) to indicate footnotes in all columns are merged and page-wise, while `\pcol@fnlayout@c` turns it *false* to make footnote typesetting column-wise, being default as well. The switch is examined by the following macros to do special operations for page-wise footnotes if it is *true*.

- `\pcol@makecol` shrinks `\@colht` and put the stretch/shrink factor of `\skip\footins` at the bottom of the column-page to be built by the macro if the page has footnotes. If the column-page is in the top page p_t , the macro also saves `\footins` into `\pcol@currfoot` which then will be saved into $\pi^f(p_t)$ by `\pcol@startpage`, or `\footins` is discarded otherwise.
- `\pcol@startcolumn`, if it is invoked from `\pcol@output` to start a column-page in a page p , `\inserts` $\pi^f(p)$ through `\footins`, and also the deferred footnotes in Φ by `\pcol@deferredfootins`, if $p = p_t$.
- `\pcol@output@switch` saves `\footins` into $\pi^f(p_t)$ if the macro is to leave a column-page in the top page, or discards it otherwise.
- `\pcol@restartcolumn` to restart a column-page in a page p `\inserts` $\pi^f(p)$ through `\footins`, and also the deferred footnotes in Φ by `\pcol@deferredfootins`, if $p = p_t$.
- `\pcol@sync` examines whether the total height of page-wise footnotes is too large to let them reside in the page to be synchronized or flushed as a whole.

- `\pcol@zparacol` redefines `\footnoterule` so that it refers to `\textwidth` rather than `\columnwidth` to determine the width of the rule above footnotes.
- `\pcol@fntext` invokes `\pcol@fntextother` to add the footnote given to it to \mathcal{F} as a deferred footnote if $p < p_t$.
- `\pcol@fntextbody` lets `\hsize` be `\textwidth` rather than `\columnwidth` to typeset the footnote given to it.

`\ifpcol@mgfnote` The switch `\ifpcol@mgfnote` is turned *true* by `\pcol@fnlayout@m` to indicate footnotes in the starting page and last page of a `paracol` environment are merged with those in pre- and post-environment stuff, while `\pcol@fnlayout@p` and `\pcol@fnlayout@c` turn it *false* to put them above/below the columns in the starting/last page respectively, being default as well. The switch is examined by the following macros to do special operations for merged page-wise footnotes if it is *true*.

- `\pcol@makenormalcol` leaves `\footins` untouched rather than putting it as a part of pre-environment stuff.
- `\pcol@makeflushedpage` leaves `\footins` untouched rather than putting it as a part of the last page of `paracol` if it works on the page.
- `\endparacol` does not let `\pcol@flushclear` examine the existence of deferred footnotes in pre-flushing column height check for the last page.

`\ifpcol@fncounteradjustment` The switch `\ifpcol@fncounteradjustment` is turned *true* by the API macro `\fncounteradjustment`, which is also invoked from `\pcol@fnlayout@p` and `\pcol@fnlayout@m` (through `\pcol@fnlayout@p`), to let `\c@footnote = b_f + n_f` by `\endparacol`. The macro `\nofncounteradjustment` turns the switch *false* to give the default state.

`\ifpcol@inner` The switch `\ifpcol@inner` is turned *false* by `\pcol@zparacol` to mean we are outside any `\vboxes`, while the macro also lets `\everyvbox` in the `paracol` environment has the operation to turn the switch *true* so that it is true whenever we are in a `\vbox`. The switch is examined by `\pcol@set@color@push` and `\pcol@icolumncolor`, the former of which also turns it *true* if we are in restricted horizontal mode, to make an `\output` request for color stack manipulation and, in the former, to reserve the stack popper by `\aftergroup`, iff the switch is *false*.

`\ifpcol@firstpage` The switch `\ifpcol@firstpage` is `\globally` turned *true* or *false* by `\pcol@output@start` if it captures pre-environment stuff as a spanning stuff or not because it is too large, respectively. Then the switch is examined by `\pcol@ioutputelt` or `\pcol@makeflushedpage` when they finds a spanning stuff to be combined to the page to be printed so as to paint the background for the color of page-wise floats unless the switch is *true* to mean the stuff is pre-environment one rather than floats. Then `\pcol@outputelt` or `\pcol@makeflushedpage` itself `\globally` turns the switch *false* after printing a page because we no longer have pre-environment stuff in the `paracol` environment we are in.

`\ifpcol@havelastpage` The switch `\ifpcol@havelastpage` is, after an initial *false*, `\globally` turned *true* by `\pcol@output@end` if it finds the last page of the `paracol` environment is connected to the post-environment stuff, or *false* otherwise. Then the switch is examined by (our own version of) `\@outputpage` which paints the background of the page to be printed iff the switch is *true* because a part of the page was produced by a `paracol` environment. Then the macro `\globally` turns the switch *false* because so far background painting should be disabled.

`\ifpcol@paired` The switch `\ifpcol@paired` is *true* if the parallel-paged typesetting should be done in *paired* mode in which the pair of left and right parallel-pages comprises a virtual page, while it is *false* if non-paired to treat the left and right pages as individual ones. Therefore, the switch is `\globally` turned *false* by `\pcol@yparacol` when `\begin{paracol}` has the optional argument for the number of columns in the left parallel-page followed by `*`, or turned *true* otherwise by `\paracol` for giving default or by `\pcol@zparacol` if it finds $C_L \geq C$ to mean parallel-paging is not in effect in reality¹³⁷.

Then the switch is examined by `\pcol@setpnoelt` and `\pcol@startpage` so that, if the switch is *false*, they let $page(q) = page(q - 1) + 2$ where $\kappa_0(\beta^p) < q \leq p_t$ in the former and $q = p_t$ in the latter, instead of $page(q) = page(q - 1) + 1$ in the usual *true* case, because the left/right pair of parallel-pages is treated as two pages rather than one.

The other macros `\pcol@ioutputelt`, (our own version of) `\@outputpage`, `\pcol@output@start`, `\pcol@imakeflushedpage`, `\pcol@iflushfloats` and `\pcol@output@end` also refer to the switch so that, if *false*, they temporarily let $\c@page = page(p) + 1$ in building the shipout image of the right component of the parallel-page pair of the page p in order to have the appropriate page number parity for the right component. Among them, `\@outputpage` has another mode dependant operation, if the switch is *true*, to decrement $\c@page$ by one before shipping out the right component to cancel the increment in the ship-out process of the left component. The macro `\pcol@addmarginpar` also examines the switch to decide the margin for marginal notes in non-paired parallel-pages. Another examiner `\pcol@zparacol` lets `\ifpcol@swapcolumn = false` only in the `paracol` environment to start if the switch is *true* because column-swapping is meaningless in non-paired parallel-paging.

`\ifpcol@swapcolumn` The switch `\ifpcol@swapcolumn`, `\ifpcol@swapmarginpar` and `\ifpcol@bg@swap` specify
`\ifpcol@swapmarginpar` that, if *true*, columns, marginal notes, and background painting in even numbered pages are
`\ifpcol@bg@swap` swapped, respectively. That is, `\ifpcol@swapcolumn` lets columns be put from right to left,
`\ifpcol@bg@@swap` `\ifpcol@swapmarginpar` lets marginal notes go to the opposite side from that in odd numbered
pages, and `\ifpcol@bg@swap` makes background painting mirrored.

Besides the initial setting to let them *false* `\globally` after the declaration, the switches are `\globally` turned *false* by `\pcol@twosided` for the cases in which API macro `\twosided` does not have ‘c’, ‘m’ or ‘b’ in its optional argument respectively, and then `\globally` turned *true* by `\pcol@twosided@k` ($k \in \{c, m, b\}$) which is invoked when the argument contains ‘c’, ‘m’ or ‘b’ respectively, or the API macro does not have the argument at all¹³⁸. The switch `\ifpcol@swapcolumn` is also turned *false* by `\pcol@zparacol` but locally if non-paired parallel-paging is specified because column-swapping is meaningless in the environment. Another modifier is (our own version of) `\@outputpage`, but setting and examining the switch in this macro is also local and is to decide the ship-out order of left and right parallel-pages.

Besides the local use by `\@outputpage`, `\ifpcol@swapcolumn` is then examined by the following macros to do special operations if it is *true* and we are in an even numbered page.

- `\pcol@swapcolumn` to reverse the order of column visiting in `\pcol@ioutputelt`, `\pcol@imakeflushedpage` and `\pcol@iflushfloats`.
- `\pcol@shiftspanning` to shift a spanning text to the left edge of text area.
- `\pcol@bg@paint@ii` to mirror the background painting of columns and column-separating gaps.

¹³⁷The initialization to let the switch *false* is not necessary because it is examined only after the first `\paracol` even in the `\@outputpage` outside `paracol` environment, but we dare to do this for the sake of clarity.

¹³⁸`\ifpcol@swapcolumn` is also turned *true* and *false* by backward compatible API macros `\swapcolumninevenpages` and `\noswapcolumninevenpages` respectively.

On the other hand, examiners of `\ifpcol@swapmarginpar` and `\ifpcol@bg@swap` are sole for each, namely `\pcol@addmarginpar` and `\pcol@bg@paint@i` respectively. If each switch is *true* and we are in an even numbered page, the former reverses `\if@firstcolumn` from the value having been set for non-swapping case, while the latter mirrors the background painting of the regions excepting columns and column-separating gaps.

A related switch `\ifpcol@bg@@swap` is let be *true* if `\ifpcol@swapcolumn` or `\ifpcol@bg@swap` is *true* and we are working on a even numbered page by `\pcol@bg@swappage` for mirrored background painting of columns and column-separating gaps, or other regions respectively, and then examined by `\pcol@bg@paintregion@i` for mirroring.

`\ifpcol@bg@painted` The switch `\ifpcol@bg@painted` is `\globally` turned *false* at the beginning of `\pcol@bg@paint@i`, and then `\globally` turned *true* by `\pcol@bg@paintregion` if it paint the region specified by its argument, i.e., `\backgroundcolor` for the region is declared. Then the switch is examined by `\pcol@bg@paint@i` to combine the painted region with others, by (our own version of) `\@outputpage` and `\pcol@outputpage@r` to incorporate painted regions into ship-out image.

`\ifpcol@bfbottom` The switch `\ifpcol@bfbottom` is *true* if `\@makecol` puts bottom floats at the bottom of a column as done by the macro in L^AT_EX's standard implementation, or *false* otherwise and thus bottom floats can be followed by footnotes as done in pL^AT_EX. Since we know only pL^AT_EX is exceptional, we let the switch *false* iff `\pfmtname` is defined and has pL^AT_EX2e in its body. The switch is examined by `\pcol@measurecolumn` to determine which footnotes or bottom floats determine D_P if both of them exist.

`\ifpcol@dfloats` The switch `\ifpcol@dfloats` is *true* iff one or more columns (in a last page) have deferred column-wise floats. Therefore, it is turned *false* by `\pcol@sync` before it invokes `\pcol@measurecolumn` which turns it *true* when it finds a column c such that $\kappa_c(\lambda_d) \neq \emptyset$. Then the switch is examined by `\pcol@makeflushedpage` to make a last page *full size*, and by `\pcol@output@end` to flush these floats.

```

14 \newif\ifpcol@output \global\pcol@outputfalse
15 \newif\ifpcol@nospan
16 \newif\ifpcol@sync \pcol@syncfalse
17 \newif\ifpcol@sptextstart \pcol@sptextstartfalse
18 \newif\ifpcol@sptext \pcol@sptextfalse
19 \newif\ifpcol@clear \pcol@clearfalse
20 \newif\ifpcol@flush
21 \newif\ifpcol@outputflt
22 \newif\ifpcol@lastpage
23 \newif\ifpcol@lastpagesave
24 \newif\ifpcol@scfnote \pcol@scfnotefalse
25 \newif\ifpcol@mgfnote \pcol@mgfnotefalse
26 \newif\ifpcol@fncounteradjustment \pcol@fncounteradjustmentfalse
27 \newif\ifpcol@inner
28 \newif\ifpcol@firstpage
29 \newif\ifpcol@havelastpage \global\pcol@havelastpagefalse
30 \newif\ifpcol@paired \global\pcol@pairedtrue
31 \newif\ifpcol@swapcolumn \global\pcol@swapcolumnfalse
32 \newif\ifpcol@swapmarginpar \global\pcol@swapmarginparfalse
33 \newif\ifpcol@bg@swap \global\pcol@bg@swapfalse
34 \newif\ifpcol@bg@@swap
35 \newif\ifpcol@bg@painted
36 \newif\ifpcol@bfbottom

```



```

37 \def\reserved@a{pLaTeX2e}
38 \ifx\reserved@a\pfmtname \pcol@bfbottomfalse \else \pcol@bfbottomtrue \fi
39 \newif\ifpcol@dfloats

```

3.3 \dimen and \skip Registers

The next declaration group is for six `\dimen` and one `\skip` registers.

`\pcol@prevdepth` The `\dimen` register `\pcol@prevdepth` is set to the depth of the last item added to the main vertical list of column c from which we switch to another column d , i.e., `\prevdepth` seen in `\pcol@invokeoutput` before `\output` request. The value of the register is then set into `\prevdepth` also by `\pcol@invokeoutput` after `\output` for the column d . The value of the register is stored in $\kappa_c(\delta)$ by `\pcol@setcurrcol` and then restored into the register by `\pcol@igetcurrcol` for the use in `\pcol@invokeoutput` above and in `\pcol@measurecolumn`, which may let the register and $\kappa_c(\delta)$ have ∞ if the column-page c is empty. The register is also updated by `\pcol@synccolumn` for empty main vertical list case, and by `\pcol@output@end` to be set into `\prevdepth` for the first vertical item of post-environment stuff.

`\pcol@colht` The `\dimen` register `\pcol@colht` has V'_p being the height of the tallest column in the last page taking `\textfloatsep` below bottom floats into account if any. The register is initialized to be $-\text{maxdimen}$ by `\pcol@sync` and then is examined and updated in `\pcol@measurecolumn` to find the tallest column. Besides the internal use of this exploration, its result is referred to by `\pcol@sync` as the threshold of pre-flushing column height check, and by `\pcol@makeflushedpage` through its argument given by `\pcol@output@end` to know the height of multi-column stuff in the last page. The macro `\pcol@makeflushedpage` also lets the register have 0 if the last page has nothing but non-merged page-wise footnote. The other usage of this register is in `\pcol@freshpage` to keep the value of `\@colht` of the page made by `\flushpage` or `\clearpage` so that it is given to $\@colroom = \kappa_c(\beta^r)$ of each column c in case a column c' s.t. $c' < c$ made another page for float columns updating `\@colht`.

`\pcol@textfloatsep` The `\dimen` (not `\skip`) register `\pcol@textfloatsep` has maxdimen if a column-page does not have synchronization points, to let top floats are inserted in usual way. If it has, the register may hold the vertical space amount inserted after top floats in a column-page instead of `\textfloatsep` so that, if a column only with top floats defines the first synchronization point, the space for floats are extended to the synchronization point. In this extension case, the register has the amount above biased by 10000 pt to distinguish the case from another case with ordinary top floats in which the register has non-biased `\textfloatsep`. In addition, if the register is less than maxdimen including a value equal to `\textfloatsep`, top floats are packed in a `\vbox` so that stretch/shrink factor of `\floatsep` cannot move synchronization points. After the default setting to be maxdimen by `\pcol@zparacol` and `\pcol@floatplacement`, the value of the register is stored in $\kappa_c(\xi)$ by `\pcol@setcurrcol` and then restored into the register by `\pcol@igetcurrcol` for the use in `\pcol@makecol`, `\pcol@combinefloats`, `\pcol@cflt`, `\pcol@output@switch`, `\pcol@measurecolumn`, `\pcol@addflhd` and `\pcol@synccolumn`.

`\pcol@lrmargin` The `\dimen` register `\pcol@lrmargin` is let have $\mu = \text{textwidth} - \text{linewidth}$ by `\pcol@zparacol`, so that `\linewidth` for column c is let have $w_c - \mu$ by `\pcol@invokeoutput`, which also sets `\parshape` if $\mu > 0$.

The other usage of this register is to have the left or right margin for background painting in the alias `\pcol@bg@leftmargin` for strict local use in `\pcol@bg@paint@i` and its descendent macros for background painting. That is, the register is aliased as `\pcol@bg@leftmargin` by `\pcol@bg@paint@i`, let have left or right margin by `\pcol@bg@swappage`, and then referred to by `\pcol@bg@pageleft`.

`\pagerim` The API `\dimen` register `\pagerim` has the size of *page rims* specified by users. Since the rims are the area for which background painting is inhibited, the register is used in area specification macros `\pcol@bg@paperwidth`, `\pcol@bg@paperheight`, `\pcol@bg@pageleft` and `\pcol@bg@pagetop`, in which the register has the negative counterpart of the specified value set by `\pcol@bg@paint@i`.

`\pcol@topskip` The `\skip` register `\pcol@topskip` keeps the value of `\topskip` at `\begin{paracol}` for the ordinary usage of `\topskip` which may have 0 in the starting and last page temporarily. After the initialization by `\pcol@zparacol`, it is referred to by `\pcol@getpinfo` for pages without spanning stuff and thus pre-environment stuff, by `\pcol@startpage` to let `\topskip` and $\pi^t(p)$ has it for non-starting page p , by `\pcol@output@start` for the second page if it finds pre-environment stuff is too large to combine with the multi-column stuff, and by `\endparacol` to recover `\topskip` for the pages following the last page.

`\belowfootnoteskip` The API `\skip` register `\belowfootnoteskip` has the amount of the space added below non-merged pre-environment footnotes. The register is initialized with the default 0pt, and then used in `\pcol@output@start` to measure the room in the first page, and in `\pcol@combinefootins` to add the space.

```
40 \newdimen\pcol@prevdepth
41 \newdimen\pcol@colht
42 \newdimen\pcol@textfloatsep
43 \newdimen\pcol@lrmargin
44 \newdimen\pagerim \pagerim\z@
45 \newskip\pcol@topskip
46 \newskip\belowfootnoteskip \belowfootnoteskip\z@
```

3.4 `\box` Registers

The next declaration group is for the following `\box` registers.

`\pcol@topfnotes` The `\box` register `\pcol@topfnotes` is the implementation of Φ to have the list of deferred footnotes. The register is made void by `\pcol@output@start` and then is made grown by `\pcol@fntexttother` with a deferred footnote added by the macro. The macro `\pcol@deferredfootins` invoked from `\pcol@startcolumn` and `\pcol@restartcolumn` tries to `\insert` the contents of the register through `\footins` but may keep some of trailing ones in it if the total height of the footnotes is too large, while `\pcol@output@end` does the `\insertion` without height capping. The macro `\endparacol` with non-merged page-wise footnote typesetting also refers to the register to pass it to `\pcol@flushclear` as its argument so as to make an explicit page break if the register has some deferred footnotes.

`\pcol@prespan` The `\box` register `\pcol@prespan` keeps the pre-spanning-text stuff during a spanning text is processed by $\text{T}_{\text{E}}\text{X}$ and our own `\output` routine. That is, the macro `\pcol@putbackmv1` saves the contents $\kappa_0(\beta^b)$ of the column 0 to be restarted into the register instead of putting it back to the main vertical list, or makes the register \perp if the column has nothing, when the restart follows the synchronization with `\ifpcol@sptextstart = true`. Then the contents of the register is put back to the main vertical list together with the box having spanning text after its vertical size is registered in the list $\pi^s(p)$ of spanning text positions and heights, by `\pcol@makecol` when the text sees a page break, or by `\pcol@output@switch` when the text is completed.

`\pcol@rightpage` The `\box` register `\pcol@rightpage` is used to build (a part of) the ship-out image of a right parallel-page in it. The macros;

`\pcol@outputelt`, `\pcol@ioutputelt`, `\pcol@makeflushedpage`,
`\pcol@flushfloats`, `\pcol@output@flush`, and `\pcol@output@clear`

work on the register together with `\@outputbox` for the left parallel-page to pass both of them to (our of version of) `\@outputpage`. The macro `\pcol@output@end` also uses the register to paint the background of the empty counterpart of non-merged page-wise footnotes in it, or to make the register \perp when it have an empty last page but with spanning stuff of page-wise floats. After closing a `paracol` environment, the contents of the register will be shipped out by `\@outputpage` invoked outside `paracol` environment when the post-environment stuff sees a page break, or referred to by `\pcol@output@start` as the pre-environment stuff in the right parallel-page. This right pre-environment stuff then will be combined with column-pages in the right parallel-page by `\pcol@ioutputelt` or `\pcol@imakeflushedpage` for shipping-out, or by the latter indirectly invoked from `\pcol@output@end` as the last right parallel-page again. Therefore the `\setbox` of the register in `\pcol@output@start`, `\pcol@makeflushedpage`, `\pcol@imakeflushedpage` and `\pcol@output@end` must be done `\global`¹³⁹.

`\pcol@colorstack@saved` The `\box` register `\pcol@colorstack@saved` is Γ_s to keep the color context Γ^c of column c until its current column-page becomes non-empty to avoid that the column-page only has coloring `\specials` for color stack establishing and rewinding to let `\pcol@ifempty` misjudge the column-page is non-empty. It is let have γ_0^c , if defined, and Γ by `\pcol@savecolorstack` invoked from `\pcol@startcolumn` and `\pcol@output@start`, and from `\pcol@restartcolumn` through `\pcol@putbackmv1` when we know or find the (re)starting column-page is empty. The macro `\pcol@putbackmv1` also makes the `\box` register \perp when the restarting column-page is not empty and thus the column-page has had coloring `\specials` for establishing color context at its beginning. Then the register is given to `\pcol@restorecst` by `\pcol@clearcst@unvbox` to put leading coloring `\specials` for establishing of the column-page when we complete it by `\pcol@opcol` or leave from it by `\pcol@output@switch`.

`\pcol@tempboxa` The `\box` register `\pcol@tempboxa` is used to have stuff temporarily as follows.

- The macro `\pcol@buildcolseprule` and its callee `\pcol@buildcse1t` builds the column-separating rule in the register for a page to be shipped out, while its contents is put into each column-separating gap by `\pcol@hfil`.
- In (our own version of) `\@outputpage`, the register has the background painting of the (left parallel-) page, which is inserted into the ship-out image by its callee `\pcol@outputpage@1` through `\everyvbox` and its contents `\pcol@outputpage@ev`.
- In `\pcol@scancst` and its callee `\pcol@iscancst` to scan Γ_r^c , Γ^c or Γ_s , the sequence of (un)coloring `\specials` to be put into the main vertical list is build in it.

`\pcol@tempboxb` The `\box` register `\pcol@tempboxb` is used in `\pcol@iscancst` to extract the top (last) element of Γ , Γ_r or Γ_s .

```
47 \newbox\pcol@topfnotes
48 \newbox\pcol@prespan \setbox\pcol@prespan\box\voidb@x
49 \newbox\pcol@rightpage \global\setbox\pcol@rightpage\box\voidb@x
50 \newbox\pcol@colorstack@saved
51 \newbox\pcol@tempboxa
52 \newbox\pcol@tempboxb
```

¹³⁹The `\global` setting in `\pcol@makeflushedpage` and `\pcol@imakeflushedpage`, together with `\@outputbox` which does not need `\global` assignment, is also required by the sake of simplicity in its implementation, incidentally.

3.5 `\insert Register Set`

The next declaration is for the following `\insert` register set.

`\pcol@colorins` The register set `\pcol@colorins` is to `\insert` a `\vbox` containing a (un)coloring `\special` for color pushing or popping, or the definition of a new default color of the current column. In order to make it sure that an `\insertion` does not affect `\pagetotal` and is given to `\output` with `\box255` containing the corresponding coloring `\special` put in the main vertical list, `\count\pcol@colorins` and `\skip\pcol@colorins` are let be 0, while `\dimen\pcol@colorins` is let be `\maxdimen` to allow a column-page to have virtually infinite number of `\insertions`.

The `\insertion` is done by `\pcol@icolumncolor` for a default color definition, `\pcol@set@color@push` for color pushing, and `\pcol@reset@color@pop` and `\pcol@reset@color@mpop` for color popping in non-math and math mode respectively. Then `\inserted` `\vboxes` are packed into `\box\pcol@colorins` and is given to `\output` as Γ_r to be scanned by `\pcol@clearcolorstack` to reform it as Γ , and then scanned by `\pcol@restorecolorstack` or saved into $\Gamma_s = \text{\pcol@colorstack@saved}$ by `\pcol@savecolorstack`. The register is also referred to by `\pcol@scancst<box>` to examine if `<box>` is this register or `\pcol@colorstack@saved`, and is made \perp by `\pcol@output@end` after the final reestablishment of the color stack.

```
53 \newinsert\pcol@colorins
54 \count\pcol@colorins\z@ \skip\pcol@colorins\z@ \dimen\pcol@colorins\maxdimen
```

3.6 `\toks Register`

The last declaration is for the following `\toks` register.

`\pcol@everyvbox` The register `\pcol@everyvbox` acts as `\everyvbox` in `paracol` environments. That is, by `\pcol@zparacol`, `\everyvbox` is made `\let`-equal to this register so that updates and references of `\everyvbox` is made to this register, while the real `\everyvbox` is let have the reference to the register and `\pcol@innertrue` to make `\ifpcol@inner = true` in every `\vbox`. Besides `\pcol@zparacol`, the register is referred to by `\pcol@restoreeveryvbox` to examine if it has been `\globally` updated, i.e., its contents is not `\pcol@dummysymbol`.

```
55 \newtoks\pcol@everyvbox
56
```

4 Logging Tools

Prior to the `\definitions` of macros to implement `paracol`'s functions, we define a few macros for debug logging.

`\pcol@ShowBox` The macro `\pcol@ShowBox` puts a logging `\message` showing the height, depth and width of the `\box` (or `\insert`) register `b`, or “(VOID)” if `b = \perp`. Then, if `b ≠ \perp`, `b`'s contents is dumped into `.log` file making overflow intentionally by putting `b` into `\box0` of null height, together with `\vskip` of 1pt if `b`'s height is 0, with setting `\vfuzz = 0`.

`\pcol@LogLevel` The macro `\pcol@LogLevel⟨l1⟩⟨l2⟩⟨l3⟩` defines the detailedness of logging done by logging macros. It invokes `\pcol@iLogLevel{⟨li⟩}{cs}` to let the following `\⟨cs⟩` be `\⟨cs⟩@⟨li⟩` where *l*_{*i*} is the `\romannumeral` representation of *l*_{*i*}.

`\pcol@Log@iii` • `\pcol@Log⟨cs⟩{m}{f}` is to log the contents of the `\insert` register *f* containing footnotes which is referred to by the macro `⟨cs⟩` in a context shown by *m*. The macro `\pcol@Log@iii` (*l*₁ = 3) logs the detailed contents of *f* by `\pcol@ShowBox`, while `\pcol@Log@ii` (*l*₁ = 2) just shows the height of *f* and `\pcol@Log@i` (*l*₁ = 1) does nothing.

`\pcol@Logstart@ii` • `\pcol@Logstart{m}` and `\pcol@Logend{m}` put logging `\message` ‘S:*m*’ and ‘E:*m*’ respectively to show the beginning and end of a procedure in the macro whose name is at the head of *m*, if *l*₂ = 2 and thus they are `\let`-equal to `\pcol@Logstart@ii` and `\pcol@Logend@ii`. If *l*₂ = 1, `\pcol@Logstart@i` and `\pcol@Logend@i` do nothing.

`\pcol@Logfn@ii` • `\pcol@Logfn{m}` puts a logging `\message` *m* whose head is the macro name for footnotes whose information such as the ordinal number of the footnote processed by the macro may be shown in *m* as well, if *l*₃ = 2 and thus it is `\let`-equal to `\pcol@Logfn@ii`. If *l*₃ = 1, `\pcol@Logfn@i` does nothing.

```

57 %% Logging Tools
58
59 \def\pcol@ShowBox#1{%
60   \ifvoid#1\message{(VOID)}%
61   \else
62     \message{(\the\ht#1+\the\dp#1)x(\the\wd#1)}%
63     {\vfuuz\z@ \showboxdepth\M \showboxbreadth\M
64     \setbox\z@\vbox to\z@{\ifdim\ht#1=\z@ \vskip1p@\fi \copy#1}}%
65   \fi}
66 \def\pcol@LogLevel#1#2#3{%
67   \pcol@iLogLevel{#1}{pcol@Log}%
68   \pcol@iLogLevel{#2}{pcol@Logstart}%
69   \pcol@iLogLevel{#2}{pcol@Logend}%
70   \pcol@iLogLevel{#3}{pcol@Logfn}}
71 \def\pcol@iLogLevel#1#2{%
72   \expandafter\let\expandafter\reserved@a
73     \csname #2@romannumeral#1\endcsname
74   \expandafter\let\csname #2\endcsname\reserved@a}
75 \def\pcol@Log@iii#1#2#3{\message{\string#1{#2%
76   (\number\pcol@page:\number\pcol@currcol/\number\pcol@toppage)}}%
77   \pcol@ShowBox#3\message{end\string#1}}
78 \def\pcol@Log@ii#1#2#3{\message{\string#1{#2%
79   (\number\pcol@page:\number\pcol@currcol/\number\pcol@toppage)}=\the\ht#3}}
80 \def\pcol@Log@i#1#2#3{}
81 \def\pcol@Logstart@ii#1{\message{S\string#1}}
82 \def\pcol@Logend@ii#1{\message{E\string#1}}
83 \def\pcol@Logstart@i#1{}
84 \def\pcol@Logend@i#1{}
85 \def\pcol@Logfn@ii#1{\message{\string#1}}
86 \def\pcol@Logfn@i#1{}
87 \pcol@LogLevel111
88

```

`\pcol@F@write` Another debugging tool is for investigating memory leak problems. Since `paracol` uses `\insert` registers for various purposes, management operations of them especially that for proper release of them are crucial for the correctness of the implementation. A source of toughness in

`\pcol@F@count`
`\pcol@F@n`
`\pcol@Fb`
`\pcol@Fe`

debugging *memory leak* caused by missing a release of a register back to `\@freelist` is that the resulting shortage is revealed long after the leakage to make it hard to find the point of the leakage.

The set of macros is to help such debugging by logging the acquire and release of `\insert` registers into a file named `\jobname.flx` associated with `\pcol@F@write` where `\jobname` is given by `\jobname`. After opened when `paracol` is loaded, the file is written by `\pcol@FF{m_a}{m_b}` with a line of the following form with text messages m_a and m_b , where $p = \text{\pcol@page}$, $c = \text{\pcol@currcol}$, $p_t = \text{\pcol@toppage}$, $\pi = \text{page}(p) = \text{\c@page}$, and $n_b = \text{\pcol@F@n}$ is the cardinality of `\@deferlist` counted by `\pcol@F@count`.

$$\langle m_a \rangle (\langle p \rangle : \langle c \rangle / \langle p_t \rangle : \langle \pi \rangle) = \langle n_b \rangle \langle m_b \rangle$$

The argument $\langle m_b \rangle$ is empty when `\pcol@FF` is invoked from `\pcol@F{m_a}` for snapshot, while $\langle m_b \rangle = \text{'<=<n_b'}$ when invoked from `\pcol@Fe{m_a}` paired by `\pcol@Fb = \pcol@F@count` by which the cardinality of `\@freelist` is given to $\langle n_b \rangle$ through `\pcol@F@n` and then `\reserved@a`. Therefore, by the pair of `\pcol@Fb` and `\pcol@Fe`, the consumption or restitution in a series of operations surrounded by the pair is logged in the file.

In the production version, the logging is disabled of course by `\letting \pcol@F` and `\pcol@Fe` be `\@gobble` and `\pcol@F` be `\relax`, while the open of `\pcol@F@write` is disabled as well by a pair of `\iffalse` and `\fi`.

```

89 \iffalse
90 \newwrite\pcol@F@write
91 \immediate\openout\pcol@F@write\jobname.flx
92 \fi
93 \def\pcol@F#1{\pcol@FF{#1}{}}
94 \def\pcol@FF#1#2{\pcol@F@count
95   \immediate\write\pcol@F@write{#1(\number\pcol@page:\number\pcol@currcol/%
96     \number\pcol@toppage:\number\c@page)=\pcol@F@n #2}}
97 \def\pcol@F@count{\@tempcnta\z@
98   \def\@elt##1{\advance\@tempcnta\@one}\@freelist
99   \xdef\pcol@F@n{\number\@tempcnta}}
100 \let\pcol@Fb\pcol@F@count
101 \def\pcol@Fe#1{\let\reserved@a\pcol@F@n \pcol@FF{#1}{<=<\reserved@a}}
102 \let\pcol@F\@gobble
103 \let\pcol@Fb\relax
104 \let\pcol@Fe\@gobble
105

```

5 \output Routine

`\pcol@ovf` Before giving the definitions of macros in `\output` routine, we define the macro `\pcol@ovf` invoked if `\@freelist` is empty on an acquisition of an `\insert` by `\@next` and thus we have to abort the execution by `\PackageError` with a message notifying the shortage. The additional help message is `\@ehb` as in `\@fltovf`. This macro is used in `\pcol@opcol`, `\pcol@startpage`, `\pcol@output@start`, `\pcol@output@switch`, `\pcol@iscancst`, `\pcol@savefootins`, `\pcol@flushcolumn`, `\pcol@synccolumn`, `\pcol@output@end` and `\pcol@icolumncolor`.

```

106 %% \output Routine
107
108 \def\pcol@ovf{%
109   \PackageError{paracol}{Too many unprocessed columns/floats}\@ehb}
110

```

`\pcol@output` The macro `\pcol@output` is the `paracol`'s version of `\output` which is let have this macro as its sole token by `\pcol@zparacol`. The structure of this macro is same as that of L^AT_EX's `\output` but the following replacements are made¹⁴⁰.

- `\@specialoutput` → `\pcol@specialoutput` to process L^AT_EX's genuine functions including the customized marginal note placement, and `paracol`'s own special output functions; starting first page, color context management, column-switching, page flushing with/without float flushing, and building the multi-column part of the last page.
- `\@makecol` → `\pcol@makecol` for a special care for current column-page having synchronization point and/or page-wise footnotes.
- `\@opcol` → `\pcol@opcol` to hold the current column-page which just has completed.
- `\@startcolumn` → `\pcol@startcolumn` to create a new column-page. The argument `\@one` is to distinguish the invocation in this macro from that in `\pcol@freshpage` so that the `\insertions` of $\pi^f(p)$ and Φ are done only when a new column-page is created with ordinary page break.

In addition, before we start the main body of `\output` routine, we add two operations for coloring. One is to make `\set@color` `\let`-equal to `\pcol@set@color`, i.e, to let it regain its original definition throughout `\output` routine, because no manipulation of color stack is necessary¹⁴¹. The other is to zero-clear the counter `\pcol@mcid` because we are definitely in the main (non-internal) vertical mode and thus all push/pop pairs of the coloring in math mode have been processed.

Further, before we start the sequence for non-special `\output` request on page breaks, we examine if `\ifpcol@output = true` to mean `\pcol@output@start` has already been invoked in order to cope with `\output` request *sneaking*. This sneaking happens when `\begin{paracol}` is at a critical position of page breaking at which the pre-environment stuff has already exceeds `\vsize` but T_EX cannot make the `\output` request for the page break at `\par` at the beginning of `\pcol@zparacol` because it sees `\penalty = 10000` due to, e.g., a sectioning command just preceding `\begin{paracol}`. In this case, the request is postponed until T_EX see a `\penalty` less than 10000 and thus it is made with some non-special `\outputpenalty` greater than `-10000` when T_EX sees the dummy request of `\penalty = -10004` in `\pcol@invokeoutput` for `\pcol@output@start`. At this timing, `\pcol@zparacol` has already let `\output` have `\pcol@output` of course but the request must be processed by original `\output` because it is made *outside* of the `paracol` environment which has just started. Therefore, if `\ifpcol@output = false`, we have to perform the operation sequence as the original `\output` does. Furthermore, we have to take care of the fact that a few our own settings related to `\output` routine has already been made in `\pcol@zparacol`, namely `\if@twocolumn = true` and `\@combinefloats = \pcol@combinefloats`, which should make the macros in the original sequence confused especially by the former. Therefore, we turns `\if@twocolumn = false` and let `\@combinefloats` have the original definition kept in `\pcol@@combinefloats`¹⁴² temporarily, i.e., only in the group automatically surrounding the invocation of `\output`.

Another addition is to assign `\@maxdepth` to `\maxdepth` in order to nullify the temporary setting to 0 done in `\@addtobot`. By this assignment, in `paracol` environments T_EX's

¹⁴⁰Besides the logging with `\pcol@Logstart` and `\pcol@Legend`.

¹⁴¹Though this operation is not necessary because `\everyvbox` should work for any `\set@color` because they should be in a `\vbox`, we dare to do it for clarity.

¹⁴²Though we know `\pcol@combinefloats` acts exactly as `\@combinefloats` because `\pcol@zparacol` initializes `\pcol@textfloatsep = ∞` and `\ifpcol@lastpage = false`. On the other hand, we don't cancel the redefinition of `\footnoterule` because it should be `\textwidth = \columnwidth` outside of `paracol` environments.

page builder always refers to the value in `\@maxdepth`. Yet another addition is to add `\ifpcol@sptextstart = false` to the condition for the warning of too short `\vsize`, because a spanning text can start near the bottom of a page with a small `\@colroom` less than $1.5 \times \text{\baselineskip}$ and thus the warning is unnecessary and inappropriate when `\ifpcol@sptextstart = true`.

```

111 \def\pcol@output{\let\par\@par \let\set@color\pcol@set@color
112 \global\pcol@mcid\z@
113 \pcol@Logstart{\pcol@output\number\outputpenalty
114 (\number\c@page:\number\pcol@currcol)}%
115 \ifnum\outputpenalty<-\@M
116 \pcol@specialoutput
117 \else\ifpcol@output
118 \pcol@makecol
119 \pcol@opcol
120 \pcol@startcolumn\@ne
121 \@whilesw\if@fcolmade\fi{\pcol@opcol \pcol@startcolumn\@ne}%
122 \else
123 \@twocolumnfalse \let\@combinefloats\pcol@@combinefloats
124 \@makecol
125 \@opcol
126 \@startcolumn
127 \@whilesw\if@fcolmade\fi{\@opcol \@startcolumn}%
128 \fi\fi
129 \global\maxdepth\@maxdepth
130 \ifnum\outputpenalty>-\@Miv
131 \ifdim\@colroom<1.5\baselineskip
132 \ifdim\@colroom<\textheight
133 \ifpcol@sptextstart
134 \global\vsize\@colroom
135 \else
136 \@latex@warning@no@line{Text page \thepage\space
137 contains only floats}%
138 \emptycol
139 \fi
140 \else
141 \global\vsize\@colroom
142 \fi
143 \else
144 \global\vsize\@colroom
145 \fi
146 \else
147 \global\vsize\maxdimen
148 \fi
149 \pcol@Logend\pcol@output}
150

```

6 Completing Column-Page

`\pcol@@makecol` The macro `\pcol@@makecol⟨d⟩` is used in `\pcol@flushcolumn` and `\pcol@imakeflushedpage` which simply require L^AT_EX's original `\@makecol` to build the ship-out image of a column-page. The reason why we need our own version is that a variation of L^AT_EX, namely pL^AT_EX for Japanese, carelessly implements its own `\@makecol` to make the resulting `\@outputbox` has a

depth larger than `\@maxdepth` if the column-page has column-wise footnotes whose last line is unusually deep. To cope with the problem, this macro at first invokes `\@makecol`, and then reshape `\@outputbox` assigning $d = \@maxdepth$ to `\boxmaxdepth` to cap its depth, unless this macro is used for the last page with $d = 0$ because depth of the last component of the `\@outputbox` is incorporated in `\@colht`.

```

151 %% Completing Column-Page
152
153 \def\pcol@makecol#1{\@makecol
154 \setbox\@outputbox\vbox to\@colht{\boxmaxdepth#1\unvbox\@outputbox}}

```

`\pcol@makecol` The macro `\pcol@makecol` is invoked solely from `\pcol@output` to build the shipping image of the current column-page which just has completed in `\@outputbox`. This macro has two additional functions to its original version `\@makecol`¹⁴³ invoked in this macro.

First, if $\kappa_c(\xi) \neq \infty$ to mean the column-page has synchronization points, `\@makecol` is invoked with a special `\@textbottom` to put a vertical skip having `1/10000fil` as its stretch and shrink. This modification is to nullify not only finite stretches (as `\raggedbottom` does) but also finite shrinks possibly inserted just below the last synchronization point to move up the first visible item upward a little bit if active. Therefore, `\flushbottom` setting is nullified for column-pages having synchronization points and a small excess from the bottom of a column-page cannot be absorbed by shrinks but visible at the bottom¹⁴⁴. Note that the original definition of `\@textbottom` is saved in `\pcol@textbottom` before the invocation of `\@makecol` and is restored after that¹⁴⁵.

Second, if `\pcol@sptext = true` and $c = 0$ to mean a spanning text encounters a page break, we have the first half (or second or succeeding part if the text lays across three or more pages) of the text in `\box255`. Therefore, we add an element $span(H_n, h_n)$ to the tail of the list of spanning texts $\pi^s(p_t) = \pcol@sptextlist$, where H_n is the height of pre-spanning-text stuff in `\pcol@prespan`¹⁴⁶ plus the total height of top floats calculated by `\pcol@addflhd` with $\kappa_c(\lambda_t) = \@toplist$, and h_n is the height-plus-depth of `\box255`. Note that H_n and h_n are represented in the form of integer and thus we produce them by expansions with `\number`.

The addition, however, is not made if $h_n = 0$ because painting its background is harmful if an extension is specified to make the region visible, while not painting or drawing a segment of column-separating rule is very natural. Note that this $h_n = 0$ case includes that in which `\box255` has nothing but its height-plus-depth is non-zero because of discarding leading skips of the spanning text as pre-break skips. This special case is detected by decapsulating `\box255` by `\unvcopy` and examining the height-plus-depth of the result¹⁴⁷. Also note that the list to be added is always for the top page, i.e., $\pi^s(p_t)$ and thus we get and update it by `\pcol@getcurrpinf` and `\pcol@defcurrpage`, because the spanning text immediately follows a synchronization point in p_t . Then we let `\box255` have the pre-spanning-text stuff followed by the spanning text being the original contents of `\box255`, which may be shifted left by $W_T - w_c = \text{textwidth} - \text{columnwidth}$ by the macro `\pcol@shiftspanning` if column-swapping is in effect so that its left edge is aligned to that of the leftmost column, i.e., of the text area.

¹⁴³Not `\pcol@makecol` because the depth capping of `\@outputbox` is done by `\pcol@opcol` when it saves the box into $\kappa_c(\beta^b)$.

¹⁴⁴That is, the author gives higher priority to the perfect alignment of the items following a synchronization point.

¹⁴⁵This save/restore cannot be done by a grouping because `\@makecol` builds `\@outputbox` by local assignments.

¹⁴⁶Since we have a synchronization point before a spanning text always, pre-spanning-text stuff or its sole contents `\vbox` has a vertical skip at its tail to make the its depth 0 as discussed in §11.7.

¹⁴⁷We cannot do `\setbox\@cclv\vbox{\unvbox\@cclv}` because it erases the effect of pre-break skip following some visible material.

The third addition is for page-wise footnotes. If they are presented in `\footins`, we shrink `\@colht` by its height plus depth by `\pcol@shrinkcolbyfn` and put the stretch and shrink factor of `\skip\footins` at the bottom of `\box255` by `\pcol@unvbox@cclv` to *remove* footnotes from the column-page but keeping the stretch/shrink contribution to the page breaking by their existence. Then we save `\footins` into a new `\insert` to be referred to as `\pcol@currfoot`¹⁴⁸ by `\pcol@savefootins` if $p = p_t$ so that it is saved in $\pi^f(p)$ by `\pcol@startpage` afterward, or simply discard the contents of `\footins` otherwise because $\pi^f(p)$ has already been fixed. Note that these saving/discarding make `\footins` void and thus `\@makecol` will not put footnotes.

On the other hand, if footnote typesetting is column-wise, `\footins` is kept unchanged so that its contents will be put by `\@makecol` if it has something. As for `\pcol@currfoot`, it should have its default value `\voidb@x = \perp` assigned to it beforehand, so that, if $p = p_t$, `\pcol@startpage` will make $\pi^f(p) = \perp$ unless page-wise footnotes are given in `\footins`.

```

155 \def\pcol@makecol{\let\pcol@textbottom\@textbottom
156 \ifdim\pcol@textfloatsep=\maxdimen\else
157 \def\@textbottom{\vskip\z@\@plus.0001fil\@minus.0001fil}\fi
158 \ifpcol@sptext \ifnum\pcol@currcol=\z@
159 \pcol@getcurrpinfo\@tempcnta\@tempdima\@tempskipa
160 \setbox\@tempboxa\vbox{\unvcopy\@cclv}%
161 \@tempdimb\ht\@tempboxa \advance\@tempdimb\dp\@tempboxa
162 \ifdim\@tempdimb>\z@
163 \@tempdimb\ht\@cclv \advance\@tempdimb\dp\@cclv
164 \dimen@ \ht\pcol@prespan \pcol@addflhd\@toplist\pcol@textfloatsep
165 \@cons\pcol@sptextlist{\number\dimen@}{\number\@tempdimb}}%
166 \fi
167 \pcol@defcurrpage{\number\@tempcnta}\pcol@spanning\pcol@footins
168 {\pcol@sptextlist}{\pcol@mparbottom}}%
169 \setbox\@cclv\vbox{\unvbox\pcol@prespan \pcol@shiftspanning\@cclv
170 \unvbox\@cclv}%
171 \fi\fi
172 \def\pcol@currfoot{\voidb@x}%
173 \ifpcol@scfnote \ifvoid\footins\else
174 \pcol@shrinkcolbyfn\@colht\footins\relax
175 \setbox\@cclv\vbox{\pcol@unvbox\@cclv\footins}%
176 \ifnum\pcol@page=\pcol@toppage
177 \pcol@Log\pcol@makecol{save}\footins
178 \pcol@Fb
179 \pcol@savefootins\pcol@currfoot
180 \pcol@Fe{makecol(pagefn)}}%
181 \else
182 \pcol@Log\pcol@makecol{discard}\footins
183 \setbox\@tempboxa\box\footins
184 \fi
185 \fi\fi
186 \pcol@Logstart\pcol@makecol
187 \ifvoid\footins\else \pcol@Log\@makecol{put}\footins \fi
188 \@makecol
189 \pcol@Logend\pcol@makecol
190 \let\@textbottom\pcol@textbottom}

```

`\@combinefloats` The macro `\pcol@combinefloats` being our own version of `\@combinefloats` is used in L^AT_EX's `\pcol@combinefloats` `\@makecol` because the original and our own are made `\let`-equal by `\pcol@zparacol`, and `\pcol@@combinefloats`

¹⁴⁸Not in `\pcol@footins` because it is destroyed in `\pcol@startpage` just before saving operation into $\pi^f(p)$.

also used in `\pcol@makenormalcol` explicitly. The customization is twofold for both of top and bottom floats.

For the top floats, we invoke the original `\cflt` if $\kappa_c(\xi) = \infty$ to mean the column-page to be shipped out does not have synchronization points, or otherwise our own `\pcol@cflt` which we will discuss shortly. Prior to the invocation of `\cflt`, in addition, we let `\maxdepth = \@maxdepth` so that the macro refers to the value used throughout a `paracol` environment instead of that modified by `\@addtobot` possibly affect the work in `\cflt` by the following sequence.

```
\pcol@flushcolumn(c) → \pcol@trynextcolumn → ⋯ → \@addtobot
                        → \pcol@flushcolumn(c+k) → \pcol@makecol → \@makecol
                        → \pcol@combinefloats → \cflt
```

For the bottom floats, we invoke the original `\cflb` always but, if the column-page has synchronization points, we insert vertical skips of $s = 0\text{pt plus }0.0001\text{fil minus }0.0001\text{fil}$ and $-s$ before and after the invocation respectively. Since `\@textbottom` is let have the skip of s by `\pcol@makecol` for a column-page having synchronization points and is inserted below bottom floats by `\@makecol`¹⁴⁹, the effect of `\@textbottom` is canceled by the skip of $-s$ but looks moved above bottom floats. Therefore, if the natural height of the column-page is smaller than `\@colht`, bottom floats are flushed to the page bottom as if column-page itself is flushed by `\newpage` etc. On the other hand, if the natural height is larger, more importantly, all shrinks below the last synchronization point is canceled by the infinite shrink in s above the bottom float but we should have sufficient space for shrinks there thanks to `\textfloatsep` to avoid the interference between the bottom text and bottom floats.

In addition, if `\ifpcol@lastpage = true` to mean the column-page is in the last page, we insert `\textfloatsep` in `\@outputbox` below the bottom floats so that they are well separated from post-environment stuff. The switch is also *true* in the invocation from `\pcol@makenormalcol` for pre-environment stuff, so that the bottom floats in it are well separated from the top of multi-column stuff in the starting page.

On the other hand, the original `\@combinefloats` saved in `\pcol@combinefloats` by `\pcol@zparacol` is used in `\pcol@output` to restore the original when it finds `\output` request sneaking.

```
191 \def\pcol@combinefloats{%
192   \global\maxdepth\@maxdepth
193   \ifx\@toplist\@empty\else
194     \ifdim\pcol@textfloatsep=\maxdimen \cflt \else \pcol@cflt \fi
195   \fi
196   \ifx\@botlist\@empty\else
197     \ifdim\pcol@textfloatsep=\maxdimen \cflb
198     \else
199       \setbox\@outputbox\vbox{\unvbox\@outputbox
200         \vskip\z@\@plus.0001fil\@minus.0001fil}%
201       \cflb
202       \setbox\@outputbox\vbox{\unvbox\@outputbox
203         \vskip\z@\@plus-.0001fil\@minus-.0001fil}%
204     \fi
205     \ifpcol@lastpage
206       \setbox\@outputbox\vbox{\box\@outputbox \vskip\textfloatsep}%
207     \fi
208   \fi}
```

¹⁴⁹The insertion point is common to L^AT_EX and pL^AT_EX.

`\pcol@cflt` The macro `\pcol@cflt` is invoked solely from `\pcol@combinefloats` if the column-page for which the macro combines the top floats has synchronization points. The macro has the same structure as L^AT_EX's version `\cflt` but has three modifications. The first one is that the floats are packed in a `\vbox` rather than listed in `\@outputbox` to nullify the stretch and shrink of `\floatsep` to keep the synchronization point from moving by them¹⁵⁰. The second is that we use `\@maxdepth` instead of `\maxdepth` to make it clear we always use the value common throughout a `paracol` environment. The third is that the `\textfloatsep` is replaced with `\pcol@textfloatsep = $\kappa_c(\xi)$` (definitely finite) which can have a value different from `\textfloatsep` when the float space is enlarged for synchronization. If this enlargement is required, $\kappa_c(\xi)$ is biased by 10000pt and thus is assuredly¹⁵¹ larger than 5000pt. If so, the insertion of `\topfigrule` should be inhibited because it has already been inserted by `\pcol@synccolumn` or there are no real floats but we only have the float for main vertical list prior to the synchronization point, or *MVL-float* in short.

```

209 \def\pcol@cflt{%
210   \let\@elt\@comflelt
211   \setbox\@tempboxa\vbox{%
212     \@toplist
213     \setbox\@outputbox\vbox{%
214       \boxmaxdepth\@maxdepth
215       \box\@tempboxa
216       \vskip-\floatsep
217       \ifdim\pcol@textfloatsep>5000\p@
218         \advance\pcol@textfloatsep-\@M\p@
219       \else
220         \topfigrule
221         \fi
222       \vskip\pcol@textfloatsep
223       \unvbox\@outputbox}%
224   \let\@elt\relax
225   \pcol@Fb
226   \xdef\@freelist{\@freelist\@toplist}%
227   \pcol@Fe{cflt}%
228   \global\let\@toplist\@empty}
229

```

`\pcol@opcol` The macro `\pcol@opcol` is invoked from `\pcol@output` for the ordinary completed column-page built by `\pcol@makecol`, or from the loop creating float columns in `\pcol@output` or `\pcol@freshpage`. At first it saves the column-page of column c in `\@outputbox`, which `\pcol@makecol` or `\@tryfcolumn` just has built for an ordinary or float column respectively, in an `\insert` acquired from `\@freelist` by `\@next`, and then adds it to the tail of $S_c = \text{\pcol@shipped-}c$ by `\@cons`. In this saving operation, we add the sequence of uncoloring `\specials` at the bottom to clear color stack by `\pcol@clearcst@unvbox` giving it `\@outputbox` to be `\unboxed` and possibly coloring `\specials` for the column-page's color context saved in Γ_s at the top, so that the succeeding column-page in printing order starts with its own color context. For this addition, furthermore, we let `\boxmaxdepth = \@maxdepth` to keep the depth capping made in the box builder from nullified¹⁵².

Then if $c = 0$, we fix the page number of the page p having the column-page and let $\pi^p(q)$ have $\text{page}(p) + (q - p)$ usually but possibly $\text{page}(p) + 2(q - p)$ with non-paired parallel-paging, for

¹⁵⁰Maybe unnecessary because of `\@textbottom` inserted by `\pcol@makecol` but ...

¹⁵¹Though not definitely in theoretical sense.

¹⁵²Or to apply the capping dropped from pL^AT_EX's `\@makecol`, or to do nothing for the box made by `\@tryfcolumn` and thus being 0 deep.

all $q \in [p+1, p_t]$ by `\pcol@setpageno`. After that, we invoke `\pcol@nextpage` to let $p = p'$ for the next column-page of c , where $p' = p + 1$ usually but can be $p + k + 1$ if we have consecutive k float pages from $p + 1$.

Next, we check if the oldest page p_b is made ready to be shipped out by the participation of the completed column-page by `\pcol@checkshipped`. If so, we invoke `\pcol@outputcolumns` giving argument 0 to ship out p_b and its successor float pages.

Finally we set up the next page p by `\pcol@startpage` if $p > p_t$ meaning it is new one, or by `\pcol@getcurrpage` otherwise, and reinitialize parameters for floats by `\pcol@floatplacement` before returning to the invoker.

```

230 \def\pcol@opcol{%
231   \pcol@Fb
232   \@next\@currbox\@freelist{\global\setbox\@currbox\vbox to\colht{%
233     \boxmaxdepth\@maxdepth
234     \pcol@clearcst@unvbox\@outputbox}}\pcol@ovf
235   \pcol@Fe{opcol}%
236   \expandafter\@cons\cename pcol@shipped\number\pcol@currcol\endcename\@currbox
237   \ifnum\pcol@currcol=\z@ \pcol@setpageno \fi
238   \pcol@nextpage
239   \pcol@checkshipped
240   \if@tempswa \pcol@outputcolumns\z@ \fi
241   \ifnum\pcol@page>\pcol@toppage \pcol@startpage
242   \else \pcol@getcurrpage
243   \fi
244   \pcol@floatplacement}
245
```

`\pcol@setpageno` The macro `\pcol@setpageno` is invoked from `\pcol@opcol` when it processes the column-page of the first column $c = 0$ to fix the page number $page(p) \leftarrow page = \c@page$ of the page $p = \pcol@page$ having the column-page. It is also invoked from `\pcol@output@switch` when it leaves from the first column to reflect a jump of `page` made in the column building. In both cases, the macro lets $\pi^p(q)$ have $page'(q) = page(p) + (q - p)$, except for the case of non-paired parallel-paging in which $page'(q) = page(p) + 2(q - p)$ instead, for all $q \in [p, p_t]$.

Since we possibly have to update $\pi(q)$ such that $q \geq p$, at first we temporarily let $\Pi^+ = (\Pi, \pi(p_t)) = \pcol@pages\pcol@currpage$ empty after copying its original value into $\Pi' = \reserved@a$. Then we scan $\pi'(q) \in \Pi'$ for all $q \in [p_b, p_t]$ by applying `\pcol@setpnoelt` to each $\pi'(q)$ giving its five components to the macro, so that the macro updates $\pi(q)$ by `\pcol@defcurrpage` letting $\pi^p(q) = page'(q)$ if $q \geq p$, or equivalently $p - q \leq 0$. Note that we let `\c@page` have $page'(q)$, but this assignment is temporary and `\c@page` will regain the value $page(p)$ after `\pcol@setpageno` finishes.

```

246 \def\pcol@setpageno{\begingroup
247   \@tempcnta\pcol@page \advance\@tempcnta-\pcol@basepage
248   \let\@elt\relax \edef\reserved@a{\pcol@pages\pcol@currpage}%
249   \global\let\pcol@pages\@empty \global\let\pcol@currpage\@empty
250   \let\@elt\pcol@setpnoelt \reserved@a
251   \endgroup}
252 \def\pcol@setpnoelt#1#2#3#4#5{%
253   {\let\@elt\relax \xdef\pcol@pages{\pcol@pages\pcol@currpage}}%
254   \ifnum\@tempcnta>\z@ \gdef\pcol@currpage{\@elt{#1}#2#3{#4}{#5}}%
255   \else \pcol@defcurrpage{\number\c@page}{#2}{#3}{#4}{#5}%
256     \advance\c@page\@ne
257   \ifpcol@paired\else \advance\c@page\@ne \fi
258   \fi
```

259 \advance\@tempcnta\m@ne}

`\pcol@defcurrpage` The macro `\pcol@defcurrpage{\pi^p(p)}{\pi^i(p)}{\pi^f(p)}{\pi^s(p)}{\pi^m(p)}` is invoked from `\pcol@makecol` to update $\pi^s(p_t)$, `\pcol@setpnoelt` to update $\pi^p(p)$, `\pcol@startpage` to initialize a newly created page, `\pcol@output@start` to initialize a starting page, `\pcol@output@switch` to update $\pi^s(p_t)$ and/or $\pi^f(p_t)$, and `\pcol@setmpbelt` to update $\pi^m(p)$. The macro `\xdefines\pcol@currpage` letting it have the page context $\pi(p)$ given by the arguments.

```
260 \def\pcol@defcurrpage#1#2#3#4#5{%
261 \let\@elt\relax \xdef\pcol@currpage{\@elt{#1}#2#3{#4}{#5}}
262
```

`\pcol@nextpage` The macro `\pcol@nextpage` is invoked solely in `\pcol@opcol` to let p be $p + k + 1$ where k is the number of float pages directly following p , i.e., $k = |\{q > p \mid p < \forall q' \leq q : \pi(q')^h < 0\}|$. For this update, the macro scans $\pi(q) \in \Pi$ for all $q \in [p_b, p_t)$ applying `\pcol@nextpelt` to $\pi(q)$, to perform the following where p_0 is p before update and $f = \text{\if@tempswa}$ being *true* at initial, to let $p \leftarrow p + k$, and then increments p to have $p + k + 1$.

$$\langle p, f \rangle \leftarrow \begin{cases} \langle p, f \rangle & q \leq p_0 \\ \langle p+1, f \rangle & q > p_0 \wedge f \wedge \pi^i(q) \neq \perp \wedge \pi^h(q) < 0 \\ \langle p, false \rangle & \text{otherwise} \end{cases}$$

```
263 \def\pcol@nextpage{\begingroup
264 \@tempcnta\pcol@page \advance\@tempcnta-\pcol@basepage
265 \@tempswatrue
266 \let\@elt\pcol@nextpelt \pcol@pages
267 \global\advance\pcol@page\@ne
268 \endgroup}
269 \def\pcol@nextpelt#1#2#3#4#5{%
270 \ifnum\@tempcnta<\z@
271 \ifvoid#2\@tempswafalse
272 \else\ifdim\dimen#2<\z@
273 \if@tempswa \global\advance\pcol@page\@ne \fi
274 \else \@tempswafalse
275 \fi\fi
276 \fi
277 \advance\@tempcnta\m@ne}
278
```

`\pcol@checkshipped` The macro `\pcol@checkshipped` is invoked solely in `\pcol@opcol` to let `\if@tempswa` be *true* iff $S_c = \text{\pcol@shipped} \cdot c \neq \emptyset$ for all $c \in [0, C)$ to mean the oldest page p_b is ready to be shipped out.

```
279 \def\pcol@checkshipped{\@tempswatrue
280 \@tempcnta\z@ \@whilenum\@tempcnta<\pcol@ncol\do{%
281 \expandafter\ifx\csname pcol@shipped\number\@tempcnta\endcsname\@empty
282 \@tempswafalse \fi
283 \advance\@tempcnta\@ne}}
284
```

`\pcol@getcurrpage` The macro `\pcol@getcurrpage` is invoked in `\pcol@opcol`, `\pcol@restartcolumn`, `\pcol@addmarginpar`, `\pcol@flushcolumn` and `\pcol@freshpage` to let;

```
\pcol@getpinfo \c@page = \pi^p(p) \colht = \pi^h(p) \topskip = \pi^t(p) \ifpcol@nospan = (\pi^i(p) = \perp)
\pcol@getcurrpinfo \pcol@spanning = \pi^i(p) \pcol@footins = \pi^f(p) \pcol@sptextlist = \pi^s(p)
\pcol@mparbottom = \pi^m(p)
```

for $p = \text{\pcol@page} \in [p_b, p_t]$. To do that, the macro scans all $\pi(q) \in \Pi^+ = (\Pi, \pi(p_t))$ applying `\pcol@getpelt` to $\pi(q) = \{\pi^p(q)\}\langle\pi^i(q)\rangle\langle\pi^f(q)\rangle\{\pi^s(q)\}\{\pi^m(q)\}$ to invoke

$$\text{\pcol@getpinfo}\{\pi^p(q)\}\langle\pi^i(q)\rangle\langle\pi^f(q)\rangle\{\pi^s(q)\}\{\pi^m(q)\}\langle pg\rangle\langle ch\rangle\langle ts\rangle$$

with the following arguments for `\global` assignments, if $q = p$.

$$\langle pg\rangle = \text{\global\c@page} \quad \langle ch\rangle = \text{\global\@colht} \quad \langle ts\rangle = \text{\global\topskip}$$

Then the macro `\pcol@getpinfo` do the obvious assignments to `\pcol@spanning`, `\pcol@footins`, `\pcol@sptextlist`, `\pcol@mparbottom` and $\langle pg\rangle$, and the following conditional assignments.

$$\langle\langle ch\rangle, \langle ts\rangle, \text{\ifpcol@nospan}\rangle = \begin{cases} \langle\text{\textheight}, \text{\pcol@topskip}, \text{true}\rangle & \pi^i(q) = \perp \\ \langle\pi^h(q), \pi^t(q), \text{false}\rangle & \pi^i(q) \neq \perp \end{cases}$$

The other macro `\pcol@getcurrpinfo` $\langle pg\rangle\langle ch\rangle\langle ts\rangle$ is invoked in `\pcol@makecol`, `\pcol@startpage`, `\pcol@output@switch`, `\pcol@sync`, `\pcol@flushcolumn` and `\pcol@makeflushedpage` do the similar assignments using `\pcol@getpinfo`, but it is not for $\pi(p)$ but for $\pi(p_t) = \text{\pcol@currpage}$. The macro `\pcol@getpinfo` also has a direct invoker `\pcol@outputelt`.

```

285 \def\pcol@getcurrpage{\begingroup
286 \@tempcnta\pcol@page \advance\@tempcnta-\pcol@basepage
287 \let\@elt\pcol@getpelt \pcol@pages\pcol@currpage
288 \endgroup}
289 \def\pcol@getpelt#1#2#3#4#5{%
290 \ifnum\@tempcnta=\z@
291 \pcol@getpinfo{#1}#2#3{#4}{#5}%
292 \global\c@page\global\@colht\global\topskip}%
293 \fi
294 \advance\@tempcnta\m@ne}
295 \def\pcol@getpinfo#1#2#3#4#5#6#7#8{\pcol@nospantrue
296 \gdef\pcol@spanning{#2}\gdef\pcol@footins{#3}\gdef\pcol@sptextlist{#4}%
297 \gdef\pcol@mparbottom{#5}%
298 #6#1\relax
299 \ifvoid#2\relax #7\textheight #8\pcol@topskip
300 \else #7\dimen#2\relax #8\skip#2\relax \pcol@nospanfalse
301 \fi}
302 \def\pcol@getcurrpinfo{%
303 \edef\reserved@a{\expandafter\@cdr\pcol@currpage\@nil}%
304 \expandafter\pcol@getpinfo\reserved@a}
305

```

`\pcol@floatplacement` The macro `\pcol@floatplacement` is invoked from `\pcol@opcol`, `\pcol@output@start`, `\pcol@flushcolumn`, `\pcol@freshpage` and `\pcol@output@end` to reinitialize the parameters of column-wise float placement at the beginning of a column-page or that of post-environment stuff. The macro lets `\@textfloatsheight` be 0 and then invokes `\@floatplacement`, as `\@opcol` does its tail¹⁵³. In addition, the macro lets `\pcol@textfloatsep` be `\maxdimen` to mean the new column-page does not have synchronization point at initial.

```

306 \def\pcol@floatplacement{%
307 \global\@textfloatsheight\z@ \global\pcol@textfloatsep\maxdimen
308 \floatplacement}
309

```

¹⁵³But `\mparbottom = 0` is not done because it is meaningless now.

7 Starting New Page

`\pcol@startpage` The macro `\pcol@startpage` is invoked from `\pcol@opcol` with `\pcol@currpage = $\pi(p-1)$` to start a new page $p = \text{\pcol@page}$, or from `\pcol@output@start` with a too large pre-environment stuff or `\pcol@freshpage` with `\pcol@currpage = {}` and `\c@page = $page(p)$` to start a new page $p = 0$.

First, we let `\pcol@firstprevdepth = \relax` to mean we have (had) left from the starting page so that `\pcol@output@end` will be informed of that. Next we let $p_t = p$ and then, if invoked from `\pcol@opcol`, obtain $\pi^p(p-1)$ by `\pcol@getcurrpinfo` to have $page(p-1)$ in `\c@page`, and let $\Pi \leftarrow \Pi^+ = (\Pi, \pi(p-1))$ with $\pi^f(p-1) = \text{\pcol@currfoot}$ into which `\pcol@makecol` saved page-wise footnotes if any. Next we let `\c@page = $page(p-1) + 1$` unless non-paired parallel-paging is in effect or in other words if `\ifpcol@paired = true`, or `\c@page = $page(p-1) + 2$` otherwise, by `\stepcounter`. Then we let `\@colht = \text{textheight}` as the base value without spanning stuff, and `\topskip = \pcol@topskip` because the new page is the second or succeeding one built in `paracol` environment.

```

310 %% Starting New Page
311
312 \def\pcol@startpage{%
313   \global\let\pcol@firstprevdepth\relax
314   \global\pcol@toppage\pcol@page
315   \ifx\pcol@currpage\@empty\else
316     \pcol@getcurrpinfo{\global\c@page}\@tempdima\@tempskipa
317     \@cons\pcol@pages
318     {\number\c@page}\pcol@spanning\pcol@currfoot
319     {\pcol@sptextlist}{\pcol@mparbottom}}%
320   \stepcounter{page}\ifpcol@paired\else \stepcounter{page}\fi
321   \fi
322   \global\@colht\textheight
323   \global\topskip\pcol@topskip

```

Then, we build float pages if any as follows. First we invoke `\@dblfloatplacement` to reinitialize the parameters for page-wise float placement. In addition, we let `\f@depth = 0` to nullify the setting `\f@depth = 1sp` possibly done by `\@dblfloatplacement` as discussed in the item-(2) of §1.8. Then we repeat `\@tryfcolumn` giving it `\@dbldeferlist` having page-wise floats not contributed to previous pages yet, while `\if@fcolmade = true` meaning it builds float pages in `\@outputbox`. For each float page, we acquire an `\insert` from `\@freelist` by `\@next` for $\pi^i(p_t)$ to let it have the followings to represent the float page.

$$\begin{aligned} \pi^p(p_t) &= \text{\c@page} & \pi^b(p_t) &= \text{\@outputbox} & \pi^h(p_t) &= -\text{\maxdimen} & \pi^t(p_t) &= \text{\pcol@topskip} \\ \pi^f(p_t) &= \perp & \pi^s(p_t) &= \emptyset & \pi^m(p_t) &= \emptyset \end{aligned}$$

We also increment p and p_t , and also `\c@page` by one or two according to `\ifpcol@paired` by `\stepcounter`, to let them have the values for the page following the float pages.

```

324   \@dblfloatplacement \let\f@depth\z@
325   \@tryfcolumn\@dbldeferlist
326   \@whiles\if@fcolmade\fi{%
327     \pcol@Fb
328     \@next\@currbox\@freelist{%
329       \global\setbox\@currbox\box\@outputbox}\pcol@ovf
330     \pcol@Fe{startpage(fcol)}}%
331   \global\dimen\@currbox-\maxdimen
332   \global\skip\@currbox\pcol@topskip

```



```

333 \cons\pcol@pages{{\number\c@page}\@currbox\voidb@x{}}}%
334 \stepcounter{page}\ifpcol@paired\else \stepcounter{page}\fi
335 \global\advance\pcol@page\@ne \global\pcol@toppage\pcol@page
336 \@tryfcolumn\@dbldeferlist}%

```

Next, we copy `\@dbldeferlist` containing page-wise floats which could not be included in float pages to `\reserved@b`, clear the list, and then scan the copied list by applying `\@sdblcolelt` to each list element to invoke `\@addtodblcol` for adding the element to `\@dbltoplist` or keeping it in `\@dbldeferlist` or `\@deferlist` depending on L^AT_EX's version, as L^AT_EX's `\@startdblcolumn` does. In addition, as discussed in item-(3) of §1.8, we also clear `\@deferlist` after saving it in `\reserved@c` prior to the scan, and then after the scan we concatenate `\@dbldeferlist` and `\@deferlist` to let the former have the result and restore `\@deferlist` from `\reserved@c`.

Then if this scan results in empty `\@dbltoplist` to mean the new page does not have any spanning stuff, we invoke `\pcol@defcurrpage` with $\pi^i(p_t) = \pi^f(p_t) = \perp$ and $\pi^s(p_t) = \pi^m(p_t) = \emptyset$ so that $\pi(p_t)$ represents a page perfectly empty.

```

337 \begingroup
338 \let\reserved@b\@dbldeferlist \let\reserved@c\@deferlist
339 \global\let\@dbldeferlist\@empty \global\let\@deferlist\@empty
340 \let\@elt\@sdblcolelt
341 \reserved@b
342 \let\@elt\relax \xdef\@dbldeferlist{\@dbldeferlist\@deferlist}%
343 \global\let\@deferlist\reserved@c
344 \endgroup
345 \ifx\@dbltoplist\@empty
346 \pcol@defcurrpage{\number\c@page}\voidb@x\voidb@x{}}}%

```

Otherwise, i.e., `\@dbltoplist` is not empty, we scan all elements in it by letting `\@elt = \@comdblflleft` to have all page-wise floats in `\@tempboxa`. Then, after returning all elements to `\@freelist`, we acquire an `\insert` from `\@freelist` to be $\pi^i(p_t)$ by `\@next` and store the contents of `\@tempboxa` in $\pi^b(p_t)$ after removing the last vertical skip `\dblfloatsep` and then adding `\dblfigrule` and the vertical skip `\dbltextfloatsep`. The other elements of $\pi(p_t)$ are set as follows to represent the page with spanning stuff which makes the height of each column `\@colht` shrunk from its initial value `\textheight` by the series of `\@addtodblcol`.

$$\pi^p(p_t) = \c@page = page(p_t) \quad \pi^h(p_t) = \@colht \quad \pi^t(p_t) = \pcol@topskip \quad \pi^f(p_t) = \perp$$

$$\pi^s(p_t) = \emptyset \quad \pi^m(p_t) = \emptyset$$

Finally, regardless of the existence of the page-wise floats, we let `\pcol@footins = \perp` to mean the top page does not have any page-wise footnotes, so far if footnote typesetting is page-wise, or never otherwise.

```

347 \else
348 \setbox\@tempboxa\ vbox{}%
349 \begingroup
350 \let\@elt\@comdblflleft
351 \@dbltoplist
352 \let\@elt\relax
353 \pcol@Fb
354 \xdef\@freelist{\@freelist\@dbltoplist}%
355 \pcol@Fe{startpage(dbltop)}%
356 \global\let\@dbltoplist\@empty
357 \pcol@Fb
358 \@next\@currbox\@freelist{\global\setbox\@currbox\ vbox{}%

```

```

359     \unvbox\@tempboxa \vskip-\dblfloatsep \dblfigrule
360     \vskip\dbltextfloatsep}}\pcol@ovf
361     \pcol@Fe{startpage(spanning)}%
362     \global\dimen\@currbox\@colht
363     \global\skip\@currbox\pcol@topskip
364     \pcol@defcurrpage{\number\c@page}\@currbox\voidb@x{}{}%
365     \endgroup
366     \fi
367     \gdef\pcol@footins{\voidb@x}
368

```

8 Shipping Page Out

`\pcol@outputcolumns` The macro `\pcol@outputcolumns⟨all⟩` is invoked from `\pcol@opcol` with $\langle all \rangle = 0$ to ship out the page p_b and float pages following it if any, or `\pcol@sync` with $\langle all \rangle = 1$ to ship out all pages in Π . It copies $\Pi = \text{\pcol@pages}$ into $\Pi' = \text{\reserved@b}$ and clear Π once to remove pages shipped out from it. Then, after initializing $f_o = \text{\if@tempswa} = true$ to ship out (the first) ordinary page and $f_f = \text{\ifpcol@outputflt} = true$ to ship out float pages (following the first page), it scans all $\pi(q) \in \Pi'$ applying `\pcol@outputelt⟨all⟩` to $\pi(q)$ to ship it out or keep it in Π .

```

369 %% Shipping Page Out
370
371 \def\pcol@outputcolumns#1{\begingroup
372   \def\@elt{\pcol@outputelt#1}\@tempswatrue \pcol@outputflttrue
373   \let\reserved@b\pcol@pages \gdef\pcol@pages{}%
374   \reserved@b
375   \endgroup}

```

`\pcol@outputelt` The macro `\pcol@outputelt⟨all⟩{ $\pi^p(q)$ }{ $\pi^i(q)$ }{ $\pi^f(q)$ }{ $\pi^s(q)$ }{ $\pi^m(q)$ }` ships out the ordinary or float page q if $f_o = true$ or $f_f = true$ respectively. After initializing `\@outputbox` to be \perp , we retrieve the page q 's information by `\pcol@getpinfo` to have $page(q)$ in `\c@page` locally, `\textheight` or $\pi^h(q)$ in $h = \text{\@tempdima}$, $f_{ns} = \text{\ifpcol@nospan} = (\pi^i(q) = \perp)$, and `\pcol@footins = $\pi^f(q)$` .

```

376 \def\pcol@outputelt#1#2#3#4#5#6{%
377   \setbox\@outputbox\box\voidb@x
378   \pcol@getpinfo{#2}#3#4{#5}{#6}\c@page\@tempdima\@tempkipa

```

Then, we do one of the followings according to h and f_o .

- $h < 0$
It means q is a float page. If $f_f = true$, we let `\@outputbox` have $\pi^b(q)$ to be shipped out, paint its background with $B_{\{F,f\}}$ by `\pcol@bg@paintbox` letting the basic height `\pcol@bg@floatheight` of the painting region $R_{\{F,f\}}$ be $H_T = \text{\pcol@bg@textheight} = \text{\textheight} + \text{\maxdepth}$, and return `\insert· $\pi^i(q)$` to `\@freelist` by `\@cons` because it is no longer necessary. Then if $C_L < C$ to mean parallel-paging is in effect, we let `\pcol@rightpage` be an empty box but paint its background too, because the right counterpart of left parallel float page should be always blank. Note that we temporarily increment $page(q)$ by one for non-paired right parallel-pages so that the painting macro performs page-parity dependent operations correctly.

On the other hand if $f_f = false$, we simply return $\pi(q)$ back to Π .

```

379 \ifdim\@tempdima<\z@
380 \ifpcol@outputflt
381 \def\pcol@bg@floatheight{\pcol@bg@textheight}%
382 \setbox\@outputbox\ vbox to\textheight{%
383 \pcol@bg@paintbox{Ff}\unvbox\pcol@spanning}%
384 \pcol@Fb
385 \@cons\@freelist\pcol@spanning
386 \pcol@Fe{outputelt(spanning)}%
387 \ifnum\pcol@ncolleft<\pcol@ncol
388 \setbox\pcol@rightpage\ vbox to\textheight{%
389 \ifpcol@paired\else \advance\c@page\@ne \fi
390 \pcol@bg@paintbox{Ff}\vfil}%
391 \fi
392 \else
393 \@cons\pcol@pages{{#2}#3#4{#5}{#6}}%
394 \fi

```

- $h \geq 0 \wedge f_o = true$

It means q is a non-float page to be shipped out. If $\langle all \rangle = 0$, we let $f_o = false$ to keep succeeding non-float pages from being shipped out. Then we build the ship-out image of the right parallel-page q in `\pcol@rightpage` by `\pcol@ioutputelt` giving it the box and the column range $[C_L, C)$ if $C_L < C$ to mean parallel-paging, and then that of the left parallel-page in `\@outputbox` by `\pcol@ioutputelt` again but giving it $[0, C_L)$ and `\@outputbox`. Note that the right-first left-second order is essential, because in the process to build right parallel-page we have to examine the existence of $\pi^i(q)$ and $\pi^j(q)$ and then refer to their height and depth to make the region corresponding to them blank, while the boxes of these `\inserts` are made void in the building process of the left parallel-page obviously.

Then after the ship-out image building, we `\globally` let `\ifpcol@firstpage = false` to tell `\pcol@ioutputelt` and `\pcol@makeflushedpage` that the pages they build are no longer first of a `paracol` environment and thus $\pi^i(q)$ should have page-wise floats rather than pre-environment stuff hereafter.

```

395 \else\if@tempswa
396 \ifnum#1=\z@ \@tempswafalse \fi
397 \ifnum\pcol@ncolleft<\pcol@ncol
398 \pcol@Logstart{\pcol@outputelt{right}}%
399 \pcol@ioutputelt\pcol@ncolleft\pcol@ncol\pcol@rightpage
400 \pcol@Logend{\pcol@outputelt{right}}%
401 \fi
402 \pcol@Logstart{\pcol@outputelt{left}}%
403 \pcol@ioutputelt\z@\pcol@ncolleft\@outputbox
404 \pcol@Logend{\pcol@outputelt{left}}%
405 \global\pcol@firstpagefalse

```

- $h \geq 0 \wedge f_o = false$

It means q is a non-float page to be kept. Therefore, we let $f_f = false$ to keep float pages following it from being shipped out. Then we return $\pi(q)$ to Π by `\@cons`.

```

406 \else
407 \pcol@outputfltfalse
408 \@cons\pcol@pages{{#2}#3#4{#5}{#6}}%
409 \fi\fi

```

Finally, if $\text{\@outputbox} \neq \perp$ to mean $\pi(q)$ is to be shipped out, we invoke \@outputpage to do it and increment p_b to let it has $q + 1$. Note that since we have let $\text{\c@page} = \text{page}(q)$, the direct and indirect references to it in \@outputpage are correctly done. Also note that the \global increment of it by \stepcounter in \@outputpage will be overridden by the \global assignment to it done by \pcol@startpage or \pcol@getcurrpage invoked from \pcol@opcol if $\langle all \rangle = 0$, or by $\text{\pcol@getcurrpinfo}$ invoked from \pcol@sync otherwise.

```

410 \ifvoid\@outputbox\else
411   \global\advance\pcol@basepage\@ne \@outputpage
412 \fi}
413

```

\pcol@ioutputelt The macro $\text{\pcol@ioutputelt}\langle C^0 \rangle \langle C^1 \rangle \langle b \rangle$ is invoked solely in \pcol@outputelt but can be done twice with $(C^0, C^1, b) = (C_L, C, \text{\pcol@rightpage})$ if parallel-paging is in effect and with $(C^0, C^1, b) = (0, C_L, \text{\@outputbox})$ always, to build the ship-out image of the right or left parallel-page q in the box b respectively.

After opening a \vbox of \textheight tall for b , at first we increment $\text{page}(q)$ by one for right parallel-page if it is non-paired, so that painting macros perform page-parity dependent operations correctly.

Next, we put materials to be shipped out in the box b as follows. First, if $\pi^f(q) \neq \perp$ to mean the page q has page-wise footnote, we paint their background with $B_{\{N,n\}}$ by \pcol@bg@paintbox letting the basic height $\text{\pcol@bg@footnoteheight}$ of the painting region $R_{\{N,n\}}$ be the height-plus-depth of $\pi^f(q)$. We do the painting at this earliest stage of the image building in order to use the left-top corner of the text area where we are now at as the origin for painting, and to let the region may be overlaid by those of columns and column-separating gaps. We also let $h = \pi^h(q) = \text{\@tempdima}$ shrunk by the height-plus-depth by $\text{\pcol@shrinkcolbyfn}$.

```

414 \def\pcol@ioutputelt#1#2#3{\setbox#3\vbox to\textheight{%
415   \ifpcol@paired\else\ifnum#1=\z@\else \advance\c@page\@ne \fi\fi
416   \ifvoid\pcol@footins\else
417     \def\pcol@bg@footnoteheight{\@elt{\ht\pcol@footins}\@elt{\dp\pcol@footins}}%
418     \pcol@bg@paintbox{Nn}%
419     \pcol@shrinkcolbyfn\@tempdima\pcol@footins\relax
420   \fi

```

Second, if $f_{ns} = false$ to mean $\pi(q)$ has spanning stuff in $\pi^b(q)$, we do one of the followings.

- If $C^0 = 0$ to mean the target is the left parallel-page, $\pi^b(q)$ is put by \unvbox , painting its background with $B_{\{F,f\}}$ by \pcol@bg@paintbox letting the basic height $\text{\pcol@bg@floatheight}$ of the painting region $R_{\{F,f\}}$ be the height-plus-depth of $\pi^b(q)$ if q is not the first page and thus $\pi^b(q)$ has page-wise floats. We also return the \insert $\pi^i(q)$ to \@freelist by \@cons .
- If $C^0 \neq 0$ to mean right parallel-page and q is the first page, \pcol@rightpage has pre-environment stuff in the right parallel-page. Therefore, we simply put the box but making its height and depth equal to those of $\pi^b(q)$, without painting because the box has already been painted, and with \nointerlineskip to prevent baseline-skip insertion below the box.
- Otherwise, i.e., $C^0 \neq 0$ and q is not the first page, we put an empty box whose height and depth equal to those of $\pi^b(q)$ by \pcol@phantom , with painting as done for floats in the left parallel-page and with \nointerlineskip .

Note that after putting spanning stuff and/or painting the background, we temporarily increment \topmargin by the height-plus-depth of $\pi^b(q)$, so that painting macros for columns,

column-separating gaps and spanning texts assume the top edge of column area as that of text area when they extend the top edges of their regions upward to the page top.

```

421 \ifpcol@nospan\else
422   \def\pcol@bg@floatheight{%
423     \elt{\ht\pcol@spanning}\elt{\dp\pcol@spanning}}%
424   \@tempdimb\ht\pcol@spanning \advance\@tempdimb\dp\pcol@spanning
425   \ifnum#1=\z@
426     \ifpcol@firstpage\else \pcol@bg@paintbox{Ff}\fi
427     \pcol@Fb
428     \@cons\@freelist\pcol@spanning \unvbox\pcol@spanning
429     \pcol@Fe{ioutputelt(spanning)}%
430   \else\ifpcol@firstpage
431     \ht\pcol@rightpage\ht\pcol@spanning
432     \dp\pcol@rightpage\dp\pcol@spanning
433     \box\pcol@rightpage \nointerlineskip
434   \else
435     \pcol@bg@paintbox{Ff}\pcol@phantom\pcol@spanning \nointerlineskip
436   \fi\fi
437   \advance\topmargin\@tempdimb
438 \fi

```

Third, we invoke `\pcol@buildcolseprule` giving it h being $\pi^h(q)$ but possibly shrunk by page-wise footnotes, the column range $[C^0, C^1)$, and `\@maxdepth` to mean q is non-last page, to draw a column-separating rule possibly broken by the spaces for spanning texts in the box `\pcol@tempboxa` and to paint the backgrounds of columns, column-separating gaps and spanning text in the box `\@tempboxa` which we put into b immediately.

Fourth, we put a `\hbox` of `\textwidth` wide having `\hboxes` of $w_c = \text{pcol@columnwidth} \cdot c$ wide containing σ_c followed by `\hss` for all $c \in [C^0, C^1)$, where $\sigma_c = \text{box} \cdot s_c(q)$ being the first element removed from S_c by `\@next` and then returned to `\@freelist` by `\@cons` if it is not \perp , or `\voidb@x` otherwise. We separate `\hboxes` of σ_c by making each `\hbox` preceded by `\pcol@@hfil` being `\relax` for the first one and the macro `\pcol@hfil` for others giving it an argument $c^g = \text{pcol@colsepid} \in \{c, c-1\}$ which we discuss shortly to put a gap of $g_{c^g} = \text{pcol@columnsep} \cdot c^g$ wide optionally having a column-separating rule and being painted.

Note that the scanning order of $c \in [C^0, C^1)$ is usually ascending of course, but is descending if column-swapping is specified and $\text{page}(q) \bmod 2 = 0$. For this ordering, we invoke `\pcol@swapcolumn\langle c' \rangle \langle c \rangle \langle C^0 \rangle \langle C^1 \rangle` to have c for the $(c' - C^0)$ -th iteration of the scanning where $c' = \text{@tempcnta}$ and $c = \text{@tempcntb}$. Another operation done by the macro is to let $c^g = c - 1$ if swapped or $c^g = c$ otherwise, because if swapped the column- c is followed by the gap which follows $c - 1$ if not swapped.

```

439 \pcol@buildcolseprule\@tempdima#1#2\@maxdepth \unvbox\@tempboxa
440 \hb@xt@\textwidth{%
441   \let\pcol@hfil\relax
442   \@tempcnta#1\relax \@whilenum\@tempcnta<#2\do{%
443     \pcol@swapcolumn\@tempcnta\@tempcntb#1#2\relax
444     \expandafter\@next\expandafter\@currbox
445     \csname pcol@shipped\number\@tempcntb\endcsname
446     \relax{\let\@currbox\voidb@x}%
447   \ifvoid\@currbox\else
448     \pcol@Fb
449     \@cons\@freelist\@currbox
450     \pcol@Fe{ioutputelt(page)}%
451   \fi
452   \expandafter\@tempdima

```

```

453     \csname pcol@columnwidth\number\@tempcntb \endcsname
454     \pcol@hfil \hb@xt@\@tempdima{\box\@currbox\hss}%
455     \edef\pcol@hfil{\noexpand\pcol@hfil{\pcol@colsepid}}%
456     \advance\@tempcnta\@ne}}%

```

Fifth, if $\pi^f(q) \neq \perp$ to mean the page q has page-wise footnotes, we put them at the bottom of $\@outputbox$ by $\pcol@putfootins$, and return $\pi^f(q)$ to $\@freelist$, if $C^0 = 0$ meaning left parallel-page. Otherwise for the right parallel-page, we simply put an empty box whose height and depth equal to those of $\pi^f(q)$ by $\pcol@phantom$, preceded by a vertical skip of $\@skip \cdot \pi^f(q)$ and then $\@nointerlineskip$ to inhibit baseline skip insertion above the box, and followed by $\@null \vskip$ as done in $\pcol@putfootins$.

Sixth and finally¹⁵⁴, we let $\@boxmaxdepth = \@maxdepth$ to cap the depth of b which we are now closing, as done for each column-page and as expected to be applied to page-wise footnotes.

```

457 \ifvoid\pcol@footins\else
458 \ifnum#1=\z@
459 \pcol@Log\pcol@outputelt{output}\pcol@footins
460 \pcol@putfootins\pcol@footins
461 \pcol@Fb
462 \@cons\@freelist\pcol@footins
463 \pcol@Fe{ioutputelt{footins}}%
464 \else
465 \vskip\skip\pcol@footins \nointerlineskip
466 \pcol@phantom\pcol@footins \vskip\z@
467 \fi
468 \fi
469 \boxmaxdepth\@maxdepth}}

```

$\pcol@phantom$ The macro $\pcol@phantom\langle b \rangle$ is used in $\pcol@ioutputelt$, $\pcol@makeflushedpage$ and $\pcol@output@end$ to put an empty box, whose height and depth are equal to that of the argument box b being a kind of page-wise stuff, into $\pcol@rightpage$ for the right parallel-page whose left counterpart has b in it. That is, the macro is used to make a region corresponding to b blank. To put the empty box, we locally let $\@tempboxa$ have it setting its height and depth to those of b and then put it.

```

470 \def\pcol@phantom#1{#{%
471 \setbox\@tempboxa\vbox{\ht\@tempboxa\ht#1\dp\@tempboxa\dp#1\box\@tempboxa}}
472

```

$\pcol@buildcolseprule$ The macro $\pcol@buildcolseprule\langle H_{n+1} \rangle\langle C^0 \rangle\langle C^1 \rangle\langle d \rangle$ is used in $\pcol@ioutputelt$, $\pcol@makeflushedpage$ and $\pcol@iflushfloats$ to build a box containing column-separating rule possibly broken by spanning texts and to paint backgrounds of columns and column-separating gaps for $c \in [C^0, C^1]$ and spanning texts in the last page ($d = 0$) or non-last page ($d = \@maxdepth$) p having column-pages of H_{n+1} tall where $n = |\pi^s(p)|$.

For initializing the drawing and painting process, we let $\@tempdimb = H_0 + h_0 = 0$, $(\pcol@bg@from, \pcol@bg@to) = (C_b^0, C_b^1) = (C^0, C^1)$, and make boxes $b_r = \pcol@tempboxa$ for the rule and $b_b = \pcol@tempboxa$ for the background empty. Then we apply $\pcol@buildcselt@S\langle H_i \rangle\langle h_i \rangle$ to each element $span(H_i, h_i)$ of $\pi^s(p)$ to under-paint the background of each spanning text by $\pcol@bg@paintbox$ defining its region $R_S(i)$ by letting their top edge positions $y_0 = \pcol@bg@spanningtop = H_i$, and height $y_1 - y_0 = \pcol@bg@spanningheight =$

¹⁵⁴Not necessary to be finally, but we placed this assignment at the end of the box to make it clear the depth capping is only for the box.

h_i if $H_i + h_i < H_{n+1}$ to mean the spanning text is non-last, or $(H_{n+1} - H_i) + d$ if last to fill the narrow strip of $d = \text{\@maxdepth}$ tall below the text for non-last pages.

Then we scan $\pi^s(p)$ again but applying $\text{\@pcol@buildcselet}\langle H_i \rangle \langle h_i \rangle$ to each element $\text{span}(H_i, h_i)$ to do the followings.

1. To b_r , add a vertical rule whose height is $H'_i = H_i - (H_{i-1} + h_{i-1})$ and width is \@columnseprule if $H'_i > 0$, and then a vertical skip of h_i , as the rule segment between $(i-1)$ -th and i -th spanning texts. Note that H_i and h_i are represented in the form of integers and thus we need sp to use them as dimensions.
2. To b_b , add painted backgrounds for all columns $c \in [C^0, C^1)$ and column-separating gaps $c \in [C^0, C^1-1)$ by $\text{\@pcol@bg@paintcolumns}$ defining their regions $R_{\{c,g\}}^c(i)$ by letting common top edge position $y_0 = \text{\@pcol@bg@columnntop} = H_{i-1} + h_{i-1}$ and common height $y_1 - y_0 = \text{\@pcol@bg@columnheight} = H'_i$, if $H'_i > 0$. Also add painted background for the i -th spanning text by $\text{\@pcol@bg@paintbox}$ as we did for under-painting but this time the region is $R_s(i)$.
3. Let $\text{\@tempdimb} = H_i + h_i$ for the next element $\text{span}(H_{i+1}, h_{i+1})$.

Then if $H'_{n+1} > 0$, we add the last rule segment of H'_{n+1} tall to b_r , and add painted backgrounds for columns and column-separating gaps as done in the step 2 above but letting $y_1 - y_0 = H'_{n+1} + d$ to let the common bottom edge of the their regions reach the bottom of text area for non-last pages.

```

473 \def\@pcol@buildcolseprule#1#2#3#4{%
474   \@tempdima#1\relax \dimen@#4\relax
475   \let\@pcol@bg@from#2\relax \let\@pcol@bg@to#3\relax
476   \setbox\@pcol@tempboxa\vbox{}\setbox\@tempboxa\vbox{}%
477   \let\@elt\@pcol@buildcselet@S \pcol@sptextlist
478   \@tempdimb\z@ \let\@elt\@pcol@buildcselet \pcol@sptextlist
479   \let\@elt\relax \advance\@tempdima-\@tempdimb
480   \ifdim\@tempdima>\z@
481     \setbox\@pcol@tempboxa\vbox{\unvbox\@pcol@tempboxa
482       \hrule\@height\@tempdima\@width\@columnseprule}%
483     \setbox\@tempboxa\vbox{\unvbox\@tempboxa
484       \let\@elt\relax
485       \edef\@pcol@bg@columnntop{\number\@tempdimb sp}%
486       \edef\@pcol@bg@columnheight{\%
487         \@elt{\number\@tempdima sp}\@elt{\number\dimen@ sp}}%
488       \pcol@bg@paintcolumns}%
489   \fi}
490 \def\@pcol@buildcselet@S#1#2{%
491   \setbox\@tempboxa\vbox{\unvbox\@tempboxa
492     \let\@elt\relax
493     \def\@pcol@bg@spanningtop{\@elt{#1sp}}%
494     \advance\@tempdima-#1sp\relax \advance\@tempdima-#2sp\relax
495     \advance\dimen@\@tempdima
496     \edef\@pcol@bg@spanningheight{\@elt{#2sp}}%
497     \ifdim\@tempdima>\z@else \@elt{\number\dimen@ sp}\fi}%
498   \pcol@bg@paintbox{S}}}
499 \def\@pcol@buildcselet#1#2{%
500   \@tempdimc#1sp \advance\@tempdimc-\@tempdimb
501   \setbox\@pcol@tempboxa\vbox{\unvbox\@pcol@tempboxa
502     \ifdim\@tempdimc>\z@ \hrule\@height\@tempdimc\@width\@columnseprule \fi

```

```

503   \vskip#2sp}%
504   \setbox\@tempboxa\vbox{\unvbox\@tempboxa
505     \let\@elt\relax
506     \edef\pcol@bg@columnstoptop{\number\@tempdimb sp}%
507     \edef\pcol@bg@columnheight{\@elt{\number\@tempdimc sp}}%
508     \ifdim\@tempdimc>\z@ \pcol@bg@paintcolumns \fi
509     \def\pcol@bg@spanningtop{\@elt{#1sp}}%
510     \advance\@tempdima-#1sp\relax \advance\@tempdima-#2sp\relax
511     \advance\dimen@\@tempdima
512     \edef\pcol@bg@spanningheight{\@elt{#2sp}}%
513     \ifdim\@tempdima>\z@\else \@elt{\number\dimen@ sp}\fi}%
514     \pcol@bg@paintbox{s}}%
515   \@tempdimb#1sp \advance\@tempdimb#2sp\relax}
516

```

`\pcol@hfil` The macro `\pcol@hfil<c>` is used in `\pcol@ioutputelt`, `\pcol@imakeflushedpage` and `\pcol@iflushfloats` to separate column $c+1$ and c or c and $c+1$ according as the columns are swapped or not in the page the caller macros are building. If `\columnseprule = r > 0`, the macro puts the followings; a horizontal space of $g_c/2 = \text{pcol@columnsep}\cdot c/2$ followed by a skip $-r/2$ to nullify the width of the rule; the rule in `\pcol@tempboxa` which `\pcol@buildcolseprule` built, with color `\pcol@colseprulecolor\cdot c` or `\pcol@colseprulecolor` according as the former is defined or not, i.e., `\colseprulecolor[mode]{color}[c]` is declared or not; and $g_c/2$ again but preceded by $-r/2$. On the other hand if $r = 0$, we simply put a space of g_c . Note that the skips of $g_c/2$ and g_c are accompanied by `1fil` infinite stretch to avoid underfull when $\sum_{c=C^0}^{C^1-2} (w_c + g_c) + w_{C^1-1} < W_T$ where $(C^0, C^1) = \{(0, C_L), (C_L, C)\}$, due to arithmetic errors in calculations of w_c and g_c ¹⁵⁵.

```

517 \def\pcol@hfil#1{%
518   \@tempdima\csname pcol@columnsep#1\endcsname\relax
519   \ifdim\columnseprule>\z@
520     \hskip.5\@tempdima\@plus1fil\relax
521     \hskip-.5\columnseprule
522     \@ifundefined{pcol@colseprulecolor#1}%
523       {\pcol@colseprulecolor}\@nameuse{pcol@colseprulecolor#1}}%
524   \copy\pcol@tempboxa \hskip-.5\columnseprule
525   \hskip.5\@tempdima\@plus1fil\relax
526 \else \hskip\@tempdima\@plus1fil\relax
527 \fi}}
528

```

`\pcol@@outputpage` The macro `\@outputpage`, being our own version of L^AT_EX's one kept in `\pcol@@outputpage`, `\@outputpage` ships out a page p or parallel-page pair in p . The reason why we redefine this macro is that we need a few special operations for parallel-paging and background painting *outside* of `paracol` environments. Therefore, the macro is not only used in our own macros `\pcol@outputelt`, `\pcol@output@start`, `\pcol@output@flush`, `\pcol@output@clear`, `\pcol@flushfloats` and `\pcol@output@end`, but also in L^AT_EX's `\@opcol` and `\@doclearpage`¹⁵⁶ invoked from our own or L^AT_EX's `\output` routine.

First we calculate $H'_M = \text{topmargin} + \text{headheight} + \text{headsep}$ to place the origin of background painting at the top edge of text area in what L^AT_EX assumes as a page, i.e., shifted 1 inch down from the real page. Then if `\ifpcol@output = true` to mean this macro is used

¹⁵⁵It is assured the sum of w_c and g_c cannot exceed W_T even with arithmetic errors and thus overfull never occurs.

¹⁵⁶And possibly in `\@outputdblcol` if double-column typesetting is done outside `paracol`.

in a `paracol` environment, we build the painted backgrounds of left and right parallel-pages in `\pcol@tempboxa = b_l` and `\@tempboxa = b_r` by putting a vertical skip of H'_M , and invoking `\pcol@bg@paintpage` with the setting $(\pcol@bg@from, \pcol@bg@to) = (C_b^0, C_b^1)$ be $(0, C_L)$ and (C_L, C) respectively. Note that `\pcol@bg@paintpage` paints backgrounds of regions $R_a^{[c]}$ for all $a \in \{T, B, L, R, C, S, t, b, l, r\}$ and $c \in [C_b^0, C_b^1)$, and for b_r we temporarily increment $page(p)$ by one if non-paired parallel-paging is in effect.

Otherwise, i.e., if `\ifpcol@output = false` indicating outside use, we build the painted backgrounds in b_l and b_r similarly but with the following differences; background painting is done if `\ifpcol@havelastpage = true` to mean the page to be shipped has the last page of closed `paracol` as its part and `\set@color \neq \relax` to mean some coloring package is loaded; page background painting is done by `\pcol@bg@paintpage` because `\pcol@bg@paintpage` is not available outside `paracol` environments; the background of post-environment stuff is painted by `\pcol@bg@paintbox` for the region $R_{\{P,p\}} = [(0, W_T)(H_B, H_T)]$ where $H_B = \pcol@bg@preposttop \in \{\pcol@bg@preposttop@left, \pcol@bg@preposttop@right\}$ having the bottom edge of the last `paracol` environment (having right parallel-page for b_r). In addition, we examine if `\pcol@rightpage \neq \perp` to mean the right parallel-page was built by `\pcol@output@end` when the last `paracol` environment was closed and, if so, make the box `\textheight` tall adding `\vfil` to its bottom.

Then regardless of `\ifpcol@output`, we do the followings; let the height and depth of b_l and b_r be 0 because they cannot occupy any real spaces in the ship-out image; temporarily let `\ifpcol@swapcolumn = false` if $page(p)$ is odd, $C_L = C$ to mean parallel-paging is not in effect¹⁵⁷, parallel-paging is done in non-paired mode¹⁵⁸, or we are outside `paracol` environments and the page does not have anything produced in environments. That is, we let `\ifpcol@swapcolumn = true` if the page has something produced by a `paracol` environment, column-swapping and parallel-paging are specified for the (last) environment¹⁵⁹, and the page number is even. Note that a page may have two or more (last pages of) `paracol` environments whose parallel-paging style can be inconsistent including the case some of them are not parallel-paged. If this inconsistency happens the page is shipped out following the style of the last environment. Also note that even if the last environment is not parallel-paged, the right parallel-page kept in `\pcol@rightpage` is assuredly shipped out.

Then if column-swapping is in effect, we ship out the right parallel-page at first by `\pcol@outputpage@r` and then the left one by `\pcol@outputpage@l` to swap the left and right. Otherwise, the ship-out order is normal and thus the invocation order is `\pcol@outputpage@l` then `\pcol@outputpage@r`. Note that if non-paired parallel-paging is in effect, the page number to given to `\pcol@outputpage@r` as its argument is $page(p) + 1$ if it is the second one, i.e., not swapped, while the argument in other cases and of `\pcol@outputpage@l` are always $page(p)$. Then finally, we `\globally` let `\ifpcol@havelastpage = false` because so far the next page does not have `paracol`'s last page especially when we are outside it, let `\pcol@bg@preposttop@left` and `\pcol@bg@preposttop@right` have 0 because, if we are outside, the next pre-environment stuff should start from the top of a page, and let $\mathcal{M} = \pcol@mparbottom@out$ be $\mathcal{M}_0 = \pcol@mparbottom@zero$ because so far we have no marginal notes given in `paracol` environments¹⁶⁰.

529 `\let\pcol@@outputpage\@outputpage`

¹⁵⁷Since the assignments of C_L and C in `\pcol@zparacol` are `\global` and they are not modified anywhere else, examining their equality outside `paracol` environments is safe and meaningful.

¹⁵⁸We need this examination because `\ifpcol@swapcolumn = false` for non-paired parallel-paging is made locally by `\pcol@zparacol`.

¹⁵⁹Column-swapping may be enabled *after* the last `paracol` environment was closed but we consider the enabling is effective for the page having the environment.

¹⁶⁰This assignment in a `paracol` environment is meaningless because \mathcal{M} is meaningless too, but not harmful.

```

530 \def\@outputpage{\begingroup
531 \@tempdima\topmargin \advance\@tempdima\headheight \advance\@tempdima\headsep
532 \ifpcol@output
533 \setbox\pcol@tempboxa\top{\vskip\@tempdima \global\pcol@bg@paintedfalse
534 \let\pcol@bg@from\z@ \let\pcol@bg@to\pcol@ncolleft
535 \pcol@bg@paintpage}%
536 \ifpcol@bg@painted \@tempwatrue \else \@tempswafalse \fi
537 \setbox\@tempboxa\top{\vskip\@tempdima \global\pcol@bg@paintedfalse
538 \ifpcol@paired\else \advance\c@page\@one \fi
539 \let\pcol@bg@from\pcol@ncolleft \let\pcol@bg@to\pcol@ncol
540 \pcol@bg@paintpage}%
541 \else
542 \def\reserved@a{\vskip\@tempdima \global\pcol@bg@paintedfalse
543 \ifpcol@havelastpage \ifx\set@color\relax\else
544 \pcol@bg@@paintpage \pcol@bg@@paintbox{Pp}%
545 \fi\fi}%
546 \setbox\pcol@tempboxa\vbox{%
547 \let\pcol@bg@preposttop\pcol@bg@preposttop@left
548 \let\pcol@bg@from\z@ \let\pcol@bg@to\pcol@ncolleft \reserved@a}%
549 \ifpcol@bg@painted \@tempwatrue \else \@tempswafalse \fi
550 \setbox\@tempboxa\vbox{\ifpcol@paired\else \advance\c@page\@one \fi
551 \let\pcol@bg@preposttop\pcol@bg@preposttop@right
552 \let\pcol@bg@from\pcol@ncolleft \let\pcol@bg@to\pcol@ncol
553 \reserved@a}%
554 \ifvoid\pcol@rightpage\else
555 \pcol@Logstart{\@outputpage{rightset}}%
556 \setbox\pcol@rightpage\vbox to\textheight{\unvbox\pcol@rightpage \vfil}%
557 \pcol@Logend{\@outputpage{rightset}}%
558 \fi
559 \fi
560 \ht\pcol@tempboxa\z@ \dp\pcol@tempboxa\z@
561 \ht\@tempboxa\z@ \dp\@tempboxa\z@
562 \ifodd\c@page \pcol@swapcolumnfalse \fi
563 \ifnum\pcol@ncolleft<\pcol@ncol\else \pcol@swapcolumnfalse \fi
564 \ifpcol@output\else \ifpcol@havelastpage\else \pcol@swapcolumnfalse \fi\fi
565 \@tempcnta\c@page
566 \ifpcol@paired\else \advance\@tempcnta\@one \pcol@swapcolumnfalse \fi
567 \ifpcol@swapcolumn \pcol@outputpage@r\c@page \pcol@outputpage@l\@tempcnta
568 \else \pcol@outputpage@l\c@page \pcol@outputpage@r\@tempcnta
569 \fi
570 \global\pcol@havelastpagefalse \gdef\pcol@bg@preposttop@left{0pt}%
571 \global\let\pcol@bg@preposttop@right\pcol@bg@preposttop@left
572 \global\let\pcol@mparbottom@out\pcol@mparbottom@zero
573 \endgroup}
574

```

`\pcol@outputpage@l` The macro `\pcol@outputpage@l`*(page)*, used solely in our own version of `\@outputpage`, at `\pcol@outputpage@r` first lets `\c@page` have *(page)* which definitely has the value that `\c@page` had when we start `\@outputpage`. That is, even when this macro is invoked after `\pcol@outputpage@r` due to swapped parallel-paging, this assignment cancels the increment of `\c@page` done in L^AT_EX's `\@outputpage` or in other word `\pcol@@outputpage` because in this case parallel-pages are paired. Then we make `\@themargin` `\let`-equal to `\evensidemargin` if two-side typesetting is in effect and `\c@page` is even, or to `\oddsidemargin` otherwise for the reference in `\pcol@outputpage@ev` as shown shortly.

Next, if background painting took place in `\@outputpage`, we let `\everyvbox` have the macro invocation `\pcol@outputpage@ev⟨bl⟩` to be expanded to the following sequence so that they are the leading materials in the `\vbox` to be `\shipout`; examination if the document is processed by a Japanese L^AT_EX named pL^AT_EX and then, if so, a control sequence `\yoko` to put materials naturally; the painted background b_l shifted right by `\@themargin`; `\nointerlineskip` to inhibit `\baselineskip` insertion after b_l ; emptying `\everyvbox` to ensure nothing will be inserted into internal `\vboxes`; and the assignment of `\yoko` to `\let` it be `\relax` if necessary. This trick with `\everyvbox` is necessary¹⁶¹ because b_l should be put *before* `\pcol@@outputpage` puts the page header, or the header would be overlaid by regions, e.g., $R_{\{t,T\}}$ in natural cases.

The tricky elements to handle `\yoko` in the sequence is necessary for pL^AT_EX whose `\@outputpage` has `\yoko` as the first element of the `\vbox` to be `\shipout`, because `\yoko` must be the first element of a box but our `\everyvbox` to put background would make it non-first. That is by the tricky elements, the `\vbox` should have `\yoko` as the first element from `\everyvbox` and then that put by pL^AT_EX's `\@outputpage` is nullified by `\let\yoko\relax` in the `\everyvbox` just for the `\vbox` to be shipped out. On the other hand in ordinary L^AT_EX, `\yoko` does not appear in the `\vbox` or is modified. The examination of the use of pL^AT_EX is also trickily done by comparing the expansion results of `\meaning\yoko` and `\string\yoko`. Since the former results in the tokens “`\yoko`” which `\string\yoko` gives us iff `\yoko` is a primitive of underlying T_EX being pT_EX if so, the comparison should give us equality iff pL^AT_EX is in use¹⁶².

Then we invoke `\pcol@@outputpage` being (p)L^AT_EX's original version of `\@outputpage` to ship out `\@outputbox` finally.

The macro `\pcol@outputpage@r⟨page⟩` performs similar operations but it does them only when `\pcol@rightpage` $\neq \perp$ to mean we are in an `paracol` environment with parallel-paging or outside it but in the page in which it resides. Other differences are as follows; $\langle page \rangle$ can be $page(p) + 1$ for non-paired right parallel-pages; `\@outputbox` is locally made `\let`-equal to `\pcol@rightpage` prior to the invocation of `\pcol@@outputpage`; and b_r is given to `\pcol@outputpage@ev` as its argument.

```

575 \def\pcol@outputpage@l#1{%
576   \pcol@Logstart{\@outputpage{left}}%
577   \global\c@page#1\relax
578   \let\@themargin\oddsidemargin
579   \if@twoside\ifodd\c@page\else \let\@themargin\evensidemargin \fi\fi
580   \if@tempswa \everyvbox{\pcol@outputpage@ev\pcol@tempboxa}\fi
581   \pcol@@outputpage
582   \pcol@Logend{\@outputpage{left}}%
583 \def\pcol@outputpage@r#1{%
584   \begingroup
585   \ifvoid\pcol@rightpage\else
586     \global\c@page#1\relax
587     \let\@outputbox\pcol@rightpage
588     \pcol@Logstart{\@outputpage{right}}%
589     \let\@themargin\oddsidemargin
590     \if@twoside\ifodd\c@page\else \let\@themargin\evensidemargin \fi\fi
591     \ifpcol@bg@painted \everyvbox{\pcol@outputpage@ev\@tempboxa}\fi
592     \pcol@@outputpage
593     \pcol@Logend{\@outputpage{right}}%
594   \fi

```

¹⁶¹Unless we rewrite `\@outputpage`.

¹⁶²Unless some other T_EX has a primitive named `\yoko`. This examination is more strict than that with `\pfmtname` for `\ifpcol@bfbottom`.

```

595 \endgroup}
596 \def\pcol@outputpage@ev#1{%
597 \edef\reserved@a{\meaning\yoko}\edef\reserved@b{\string\yoko}%
598 \ifx\reserved@a\reserved@b \yoko\fi
599 \moveright{\themargin\box#1\nointerlineskip \everyvbox{}}%
600 \ifx\reserved@a\reserved@b \let\yoko\relax \fi}
601

```

9 Starting New Column-Page

`\pcol@startcolumn` The macro `\pcol@startcolumn{f}` is invoked from `\pcol@output` with $f = 1$ and `\pcol@freshpage` with $f = 0$ to start a new column-page. This macro has two additional functions to L^AT_EX's `\@startcolumn`, one for page-wise footnotes and the other for coloring.

First, if the page p in which the new column-page resides has page-wise footnotes in $\pi^f(p) = \text{\pcol@footins}$ because the column is not the leading one, we temporarily shrink `\@colht` and `\@colroom` by the space required to put $\pi^f(p)$ by `\pcol@shrinkcolbyfn` during the trial of deferred float placement, remembering the existence of the footnotes by letting `\@tempdimb = -\skip\pcol@footins` which should be 0 otherwise. This shrinkage is essentially required when $p < p_t$ because $\pi^f(p)$ has been fixed to be a part of p and thus deferred floats cannot push footnotes down to succeeding pages. In the case of $p = p_t$, the shrinkage is also desirable to avoid unnecessary pushing down of footnotes which T_EX has decided to be in p .

Then after trying put deferred floats in the column-page by `\@tryfcolumn` and `\pcol@trynextcolumn` as done in L^AT_EX's `\@startcolumn`, we `\insert` $\pi^f(p)$, if it is has some footnotes, by letting `\footins` have it by `\pcol@getcurrfoot` so that T_EX will be aware of the footnotes when it examines the page break of the column-page. That is, if $p < p_t$ the `\insertion` is to keep the vertical space for $\pi^f(p)$ in the building process of the column-page in p because any page-wise footnotes cannot be added to p any more, and thus $\pi^f(p)$ is preserved until the page p is shipped out. On the other hand if $p = p_t$, page-wise footnotes in p can grow further and thus `\inserted` footnotes will be captured again by `\pcol@output@switch` or `\pcol@startpage`. Therefore, if $p = p_t$, we release $\pi^f(p)$ to `\@freelist`.

Then if $p = p_t$, we also `\insert` deferred footnotes in Φ until their total height reaches `\@colht` by `\pcol@deferredfootins` if $f = 1$ to mean this macro is invoked from `\pcol@output`¹⁶³. Note that the deferred footnote `\insertion` in the case of $f = 0$ will be done afterward when `\pcol@freshpage` does `\pcol@restartcolumn` at its tail. Also note that `\pcol@deferredfootins` examines if `\@tempdimb = 0` to mean $\pi^f(p) = \perp$ and thus `\skip\footins` should be taken into account in its extraction of the footnotes from Φ .

Then after restoring `\@colht` and canceling the temporary shrinkage of `\@colroom`, we invoke `\pcol@savestack` to save column-page's color context into Γ_s so that coloring `\specials` to reestablish Γ_s will be put at its top if it has something when we leave it.

```

602 %% Starting New Column Page
603
604 \def\pcol@startcolumn#1{%
605 \@tempdima\@colht \@tempdimb\z@
606 \ifvoid\pcol@footins\else
607 \pcol@shrinkcolbyfn\@colht\pcol@footins\@tempdimb
608 \fi
609 \global\@colroom\@colht
610 \@tryfcolumn\@deferlist

```

¹⁶³The `\insertion` of $\pi^f(p)$ also requires $f = 1$ but this examination is redundant because $\pi^f(p) = \perp$ definitely if $f = 0$.

```

611 \if@fcolmade\else
612   \pcol@trynextcolumn
613   \ifpcol@scfnote \ifnum#1>\z@
614     \ifvoid\pcol@footins\else
615       \edef\pcol@currfoot{\pcol@footins}%
616       \pcol@getcurrfoot\copy
617       \pcol@Log\pcol@startcolumn{insert}\footins
618       \insert\footins{\unvbox\footins}%
619       \ifnum\pcol@page=\pcol@toppage
620         \pcol@Fb
621         \@cons\@freelist\pcol@footins
622         \pcol@Fe{startcolumn(pagefn)}%
623       \fi
624     \fi
625     \ifnum\pcol@page=\pcol@toppage
626       \pcol@deferredfootins\pcol@startcolumn \fi
627   \fi\fi
628 \fi
629 \advance\@tempdima-\@colht
630 \global\advance\@colroom\@tempdima
631 \global\advance\@colht\@tempdima
632 \pcol@savecolorstack}

```

`\pcol@trynextcolumn` The macro `\pcol@trynextcolumn` is invoked from `\pcol@startcolumn` and `\pcol@flushcolumn` to try to move deferred floats in `\@deferlist` into `\@toplist` or `\@botlist`. The body of this macro is perfectly equivalent to the `\else` part of `\if@fcolmade` in L^AT_EX's `\@startcolumn`.

```

633 \def\pcol@trynextcolumn{\begingroup
634   \let\reserved@b\@deferlist
635   \global\let\@deferlist\@empty
636   \let\@elt\@scolelt
637   \reserved@b
638   \endgroup}
639

```

10 Background Painting

`\pcol@bg@from` The control sequence pair $(\pcol@bg@from, \pcol@bg@to) = (C_b^0, C_b^1)$ are made `\let`-equal to `\pcol@bg@to` $(0, C_L)$ or (C_L, C) by `\pcol@buildcolseprule` and `\@outputpage` for background painting of columns and column-separating gaps, and referred to by column scanning loops in `\pcol@bg@paint@ii` and `\pcol@bg@columnleft`. The control sequence `\pcol@bg@to` is also referred to by `\pcol@bg@paint@i` to decrement it by one temporarily so that the loop in `\pcol@bg@paint@ii` scans $c \in [C_b^0, C_b^1 - 1)$ rather than $[C_b^0, C_b^1)$. Since this decrement is done whenever a painting macro is used regardless some setting of C_b^1 , `\pcol@bg@to` has default setting with C to avoid unbound reference at the decrement. Note that since this decrement is done in a `\vbox` and an appropriate setting must have been done if C_b^1 is referred in `\pcol@bg@paint@ii`, this decrement and default setting are safe.

```

640 %% Background Painting
641
642 \let\pcol@bg@to\pcol@ncol

```

`\pcol@bg@paintpage` The macros `\pcol@bg@@paintpage`, `\pcol@bg@@paintcolumns` and `\pcol@bg@@paintbox{A}` are made `\let`-equal to their interface counterparts `\pcol@bg@paintpage`, `\pcol@bg@paintcolumns` and `\pcol@bg@paintbox` by `\pcol@zparacol` if some coloring package has been loaded. Otherwise, these interface macros are `\let`-equal to `\relax` for first two and `\gobble` for the last, so that macros in `\output` routine freely use them unaware of coloring capability. One exception is in `\@outputpage` which uses `\pcol@bg@@paintpage` and `\pcol@bg@@paintbox` explicitly when it is outside `paracol` environments, examining the availability of coloring.

The macro `\pcol@bg@paintpage` and `\pcol@bg@@paintpage` are used in `\@outputpage` to paint backgrounds of regions $R_a^{[c]}$ for all $a \in \{T, B, L, R, G, C, t, b, l, r\}$ and $c \in [C_b^0, C_b^1)$ for $a = C$ while $c \in [C_b^0, C_b^1 - 1)$ for $a = G$. Therefore, the macro invokes `\pcol@bg@paint@i` with two `\pcol@bg@paint@ii{A_b}{A_g}{A_c}`, letting $A_b = \text{TBLR}$, $A_g = \text{G}$ and $A_c = \text{C}$ in the first invocation and then $A_b = \text{tblr}$ and $A_g = A_c = \emptyset$ in the second.

The macro `\pcol@bg@paintcolumns` is used in `\pcol@buildcolseprule` and `\pcol@buildcselt` to paint backgrounds of regions $R_g^c(i)$ for $c \in [C_b^0, C_b^1 - 1)$ and $R_c^c(i)$ for $c \in [C_b^0, C_b^1)$. Therefore, the macro invokes `\pcol@bg@paint@i` with `\pcol@bg@paint@ii` giving it $A_b = \emptyset$, $A_g = \text{g}$ and $A_c = \text{c}$.

The macro `\pcol@bg@paintbox{A}` is used in the following macros with A shown in the parentheses to paint the backgrounds of regions $R_{\{a_1, a_2\}}$ where $(a_1, a_2) \in \{(S, s), (F, f), (N, n), (P, p)\}$.

```

\pcol@outputelt (Ff), \pcol@ioutputelt (Nn, Ff), \pcol@buildcselt (Ss),
\pcol@output@start (Pp), \pcol@output@clear (Ff),
\pcol@makeflushedpage (Ff), \pcol@imakeflushedpage (Nn),
\pcol@output@end (Nn).

```

The macro `\@outputpage` also uses the function but `\pcol@bg@@paintbox` explicitly with $A = \text{Pp}$. Therefore `\pcol@bg@@paintbox` invokes `\pcol@bg@paint@i` with `\pcol@bg@paint@ii` giving it $A_b = A$ and $A_g = A_c = \emptyset$.

```

643 \def\pcol@bg@@paintpage{%
644   \pcol@bg@paint@i{%
645     \pcol@bg@paint@ii{TBLR}{G}{C}\pcol@bg@paint@ii{tblr}{}}
646 \def\pcol@bg@@paintcolumns{\pcol@bg@paint@i{\pcol@bg@paint@ii}{g}{c}}
647 \def\pcol@bg@@paintbox#1{\pcol@bg@paint@i{\pcol@bg@paint@ii{#1}}}}
648

```

`\pcol@bg@paint@i` The macro `\pcol@bg@paint@i{body}` is used in `\pcol@bg@@paintpage`, `\pcol@bg@@paintcolumns` and `\pcol@bg@@paintbox` to paint backgrounds by a sequence of `\pcol@bg@paint@ii` specified in $\langle body \rangle$. The painted background is built in `\@tempboxa` being a `\vtop` having a null `\vskip` as its first element so that everything put in the box is below its reference point at its top. Then, before invoking `\pcol@bg@paint@ii` in $\langle body \rangle$, we do the followings; `\globally` let `\ifpcol@bg@painted = false` to indicate so far any painted background are produced; make `\pcol@bg@leftmargin` `\let`-equal to `\pcol@lrmargin` to use this `\dimen` register locally with the more appropriate alias; negate `\pagemargin` locally to calculate H_T easily; decrement C_b^1 by one locally for the column scanning loop for $c \in [C_b^0, C_b^1 - 1)$ in `\pcol@bg@paint@ii`; and `\offinterlineskip` to inhibit inter-line `\baselineskip` insertion in the box. Then after the invocation of the sequence of `\pcol@bg@paint@ii` in $\langle body \rangle$ and closing the box, we let the height, depth and width of the box be 0 so that it does not occupy any real space in the outer box in which the box is put. Finally, if `\ifpcol@bg@painted = true` meaning that some painted backgrounds are built in the box, we put the box into the outer box surrounding it by `\nointerlineskip` to inhibit inter-line `\baselineskip` insertion before and after it.

```

649 \def\pcol@bg@paint@i#1{%

```

```

650 \setbox\@tempboxa\top{\vskip\z@
651 \global\pcol@bg@paintedfalse
652 \let\pcol@bg@leftmargin\pcol@lmargin
653 \pagerim-\pagerim \advance\pcol@bg@to\m@ne
654 \offinterlineskip #1}%
655 \ht\@tempboxa\z@ \dp\@tempboxa\z@ \wd\@tempboxa\z@
656 \ifpcol@bg@painted \nointerlineskip \box\@tempboxa \nointerlineskip \fi}

```

`\pcol@bg@paint@ii` The macro `\pcol@bg@paint@ii{ A_b }{ A_g }{ A_c }` appears only in the argument of `\pcol@bg@paint@i` used in `\pcol@bg@@paintpage`, `\pcol@bg@@paintcolumns` and `\pcol@bg@@paintbox` to paint backgrounds of regions R_a for $a \in A_b \subseteq \{T, B, L, R, S, F, N, P, t, b, l, r, s, f, n, p\}$, R_a^c for $a \in A_g \subseteq \{G, g\}$ and $c \in [C_b^0, C_b^1 - 1)$, and R_a^c for $a \in A_c \subseteq \{C, c\}$ and $c \in [C_b^0, C_b^1)$.

First we invoke `\pcol@bg@swappage` with `\ifpcol@bg@swap` to let `\pcol@bg@leftmargin` and `\ifpcol@bg@@swap` have values according to `\ifpcol@bg@swap`, `\if@twoside` and the parity of $page(p)$. Then we invoke `\pcol@bg@paintregion(a)(c)` for all $a \in A_b$ and $c = -1$ to paint the background of R_a . Second, we invoke `\pcol@bg@swappage` again but with `\ifpcol@swapcolumn` instead of `\ifpcol@bg@swap`, and `\pcol@bg@paintregion` as well but for $a \in \{A_g, A_c\}$ and $c \in [C_b^0, C_b^1 - 1)$. Third and finally, we make yet another invocation of `\pcol@bg@paintregion` for $a \in A_c$ and $c = C_b^1 - 1$.

```

657 \def\pcol@bg@paint@ii#1#2#3{%
658 \pcol@bg@swappage\ifpcol@bg@swap\fi
659 \@tfor\reserved@b:=#1\do{\pcol@bg@paintregion\reserved@b\m@ne}%
660 \pcol@bg@swappage\ifpcol@swapcolumn\fi
661 \@tfor\reserved@b:=#2#3\do{%
662 \pcol@currcol\pcol@bg@from \@whilenum\pcol@currcol<\pcol@bg@to\do{%
663 \pcol@bg@paintregion\reserved@b\pcol@currcol
664 \advance\pcol@currcol\@ne}}%
665 \@tfor\reserved@b:=#3\do{\pcol@bg@paintregion\reserved@b\pcol@currcol}}

```

`\pcol@bg@swappage` The macro `\pcol@bg@swappage<if>\fi` is used solely in `\pcol@bg@paint@ii` but twice with `<if> = \ifpcol@bg@swap` and then with `<if> = \ifpcol@swapcolumn`, to let `\pcol@bg@leftmargin` and `\ifpcol@bg@@swap` have values for mirroring according to the truth values of `<if>` and `\if@twoside` and the parity of $page(p)$ of the page p for which background painting is taking place. That is, they are let have the values as follows.

$$\begin{aligned}
W &= \begin{cases} \oddsidemargin & page(p) \bmod 2 = 1 \vee \if@twoside = false \\ \evensidemargin & page(p) \bmod 2 = 0 \wedge \if@twoside = true \end{cases} \\
&(\pcol@bg@leftmargin, \ifpcol@bg@@swap) \\
&= \begin{cases} (W, false) & page(p) \bmod 2 = 1 \vee \langle if \rangle = false \\ (W_P - (W + W_T + 2 \text{ in}), true) & page(p) \bmod 2 = 0 \wedge \langle if \rangle = true \end{cases}
\end{aligned}$$

Note that $W_P - (W + W_T + 2 \text{ in})$ means the right margin width minus 1 in with given left margin width W , and thus $W_M = \pcol@bg@leftmargin + 1 \text{ in}$ gives us the right margin width we need in mirrored background painting.

```

666 \def\pcol@bg@swappage#1#2{%
667 \pcol@bg@leftmargin\oddsidemargin \pcol@bg@@swappfalse
668 \ifodd\c@page\else
669 \if@twoside \pcol@bg@leftmargin\evensidemargin \fi
670 #1% \ifpcol@{bg@swap,swapcolumn}
671 \pcol@bg@@swaptrue
672 \advance\pcol@bg@leftmargin\textwidth \advance\pcol@bg@leftmargin2in
673 \advance\pcol@bg@leftmargin-\paperwidth

```

```

674 \pcol@bg@leftmargin-\pcol@bg@leftmargin
675 #2% \fi
676 \fi}
677

```

`\pcol@bg@paintregion` The macro `\pcol@bg@paintregion⟨a⟩⟨c⟩` is used only in `\pcol@bg@paint@ii{A_b}{A_g}{A_c}` but as many times as $|A_b| + |A_g|(C_b^1 - C_b^0 - 1) + |A_c|(C_b^1 - C_b^0)$ to paint background region $R_a^{[c]}$ specified by `\pcol@bg@⟨a⟩` with color $B_a^c = \text{\pcol@bg@color@⟨a⟩⟨c⟩}$ or, if it is undefined, $B_a = \text{\pcol@bg@color@⟨a⟩}$.

If R_a^c or R_a is defined, the painted background is built in `\@tempboxa` with `\vtop` having null vertical skip at its top and by `\pcol@bg@paintregion@i{F_x}{F_y}{F_w}{F_h}` where the arguments are defined in the body of the macro `\pcol@bg@⟨a⟩` and thus we need triple `\expandafter` to give them to the macro. Prior to the invocation of the macro, we let `\reserved@a = a'` have $a \cdot c$ if B_a^c is defined for $a \in \{G, C, g, c\}$, or a otherwise definitely for $a \notin \{G, C, g, c\}$.

Then `\pcol@bg@paintregion@i` calculates $x_0 = \text{\@tempdima}$, $y_0 = \text{\@tempdimb}$, $x_1 = \text{\@tempdimc}$ and $y_1 = \text{\dimen@}$ of the region $R_a^{[c]}$ by `\pcol@bg@calculate⟨z⟩⟨z_0⟩{F}` giving it $(z, z_0, F) \in \{(x_0, 0, F_x), (y_0, 0, F_y), (x_1, x_0, F_w), (y_1, y_0, F_h)\}$, where (x_0, y_0) and (x_1, y_1) is the left-top and right-bottom corner of the painting region in the text-area coordinate, i.e., left-right and top-down coordinate whose origin is at the left-top corner of the left-most column. Next we modify $\{x, y\}_{\{0,1\}}$ for extension by `\pcol@bg@addext⟨z⟩{s}{d}` with $(z, s, d) \in \{(x_0, '-', 1), (y_0, '-', t), (x_1, \emptyset, r), (y_1, \emptyset, b)\}$.

Now we have $[(x_0, y_0)(x_1, y_1)]$ and thus, if not mirrored, we place $R_a^{[c]}$ at (x_0, y_0) by a vertical skip of y_0 and shifting a `\hbox` for the region right by x_0 by `\moveright`, and paint the box putting a `\vrule` of $(x_1 - x_0)$ wide and $(y_1 - y_0)$ tall, letting `\current@color` have $B_r^{[c]} = \text{\pcol@bg@color@⟨a⟩}$ and then invoking `\pcol@set@color` being the original definition of `\set@color`. On the other hand if mirroring is to be done, the region should be $[(W_T - x_1, y_0)(W_T - x_0, y_1)]$ and thus the shift amount for `\moveright` of the `\hbox` is $(W_T - x_1)$.

Then after `\pcol@bg@paintregion@i` finishes its work, `\pcol@bg@paintregion` lets the switch `\ifpcol@bg@painted = true` because we painted $R_a^{[c]}$, lets the height, depth and width of `\@tempboxa` be 0 to make it a phantom, and then put it into the outside box opened by `\pcol@bg@paint@i`. On the other hand, if neither B_r^c nor B_r defined to mean the background painting of the region is not specified, we do nothing.

```

678 \def\pcol@bg@paintregion#1#2{%
679 \@ifundefined{pcol@bg@color@#1@number#2}%
680 {\def\reserved@a{#1}}{\edef\reserved@a{#1@number#2}}%
681 \@ifundefined{pcol@bg@color@reserved@a}\relax
682 {\setbox\@tempboxa\vtop{\vskip\z@
683 \expandafter\expandafter\expandafter
684 \pcol@bg@paintregion@i\csname pcol@bg@#1\endcsname}%
685 \global\pcol@bg@paintedtrue
686 \ht\@tempboxa\z@ \dp\@tempboxa\z@ \wd\@tempboxa\z@ \box\@tempboxa}}
687 \def\pcol@bg@paintregion@i#1#2#3#4{%
688 \pcol@bg@calculate\@tempdima\z@{#1}%
689 \pcol@bg@calculate\@tempdimb\z@{#2}%
690 \pcol@bg@calculate\@tempdimc\@tempdima{#3}%
691 \pcol@bg@calculate\dimen@\@tempdimb{#4}%
692 \pcol@bg@addext\@tempdima{-}{1}\pcol@bg@addext\@tempdimc}{r}%
693 \pcol@bg@addext\@tempdimb{-}{t}\pcol@bg@addext\dimen@}{b}%
694 \vskip\@tempdimb
695 \ifpcol@bg@swap
696 \advance\@tempdima-\@tempdimc \@tempdima-\@tempdima

```



```

697 \advance\@tempdimc-\textwidth \@tempdimc-\@tempdimc
698 \moveright\@tempdimc\hbox{%
699 \advance\dimen@-\@tempdimb
700 \edef\current@color{\@nameuse{pcol@bg@color@\reserved@a}}\pcol@set@color
701 \vrule\@width\@tempdima\@height\dimen@}%
702 \else
703 \moveright\@tempdima\hbox{%
704 \advance\@tempdimc-\@tempdima \advance\dimen@-\@tempdimb
705 \edef\current@color{\@nameuse{pcol@bg@color@\reserved@a}}\pcol@set@color
706 \vrule\@width\@tempdimc\@height\dimen@}%
707 \fi}
708

```

`\pcol@bg@calculate` The macro `\pcol@bg@calculate⟨z⟩⟨z0⟩⟨F⟩` is used in `\pcol@bg@paintregion@i` and `\pcol@bg@addext` to accumulate dimensional values specified in F into a `\dimen` register z with initial value z_0 . The specification F is a sequence of `\@elt⟨f⟩` to add f to z , `\pcol@bg@negative⟨F-⟩` to subtract the amount specified by F^- from z , or macros expanded to either of them.

`\pcol@bg@dimen` The macro makes `\pcol@bg@dimen \let`-equal to z and `\@elt` to `\pcol@bg@advance`, lets $z = z_0$, and then does what is specified in F . Therefore, `\@elt⟨f⟩` appearing directly or indirectly in F does `\advance⟨z⟩⟨f⟩` for the accumulation. On the other hand, `\pcol@bg@negative` makes `\@elt \let`-equal to `\pcol@bg@nadvance` to let `\@elt⟨f⟩` do `\advance⟨z⟩-⟨f⟩` for subtraction, does F^- , and then remake `\@elt = \pcol@bg@advance`. Note that f may be expanded to a negative amount having ‘-’ its beginning to results in `\@elt⟨f⟩` expanded to `\advance⟨z⟩--⟨f'⟩` with some positive amount f' , but this double negation is legitimate in T_EX and is equivalent to `\advance⟨z⟩⟨f'⟩`. The macro `\pcol@bg@negative` is used in the following macros.

```

\pcol@bg@ext@inf@l, \pcol@bg@ext@inf@r, \pcol@bg@ext@inf@t, \pcol@bg@ext@t,
\pcol@bg@ext@b, \pcol@bg@ext@l, \pcol@bg@ext@r, \pcol@bg@ext@n, \pcol@bg@ext@p.

```

```

709 \def\pcol@bg@calculate#1#2#3{\let\pcol@bg@dimen#1\relax
710 \let\@elt\pcol@bg@advance \pcol@bg@dimen#2\relax #3}
711 \def\pcol@bg@negative#1{\let\@elt\pcol@bg@nadvance #1\relax
712 \let\@elt\pcol@bg@advance}
713 \def\pcol@bg@advance#1{\advance\pcol@bg@dimen#1\relax}
714 \def\pcol@bg@nadvance#1{\advance\pcol@bg@dimen-#1\relax}
715

```

`\pcol@bg@addext` The macro `\pcol@bg@addext⟨z⟩⟨s⟩⟨d⟩` is used only in `\pcol@bg@paintregion@i` but four times with $(z, s, d) \in \{(x_0, '-', l), (y_0, '-', t), (x_1, \emptyset, r), (y_1, \emptyset, b)\}$, to perform extension on a `\dimen` register z .

`\pcol@bg@ext@inf@t` First the macro gets $e = \pcol@bg@ext@d \cdot a' \in e_a^{[c]}(\{x, y\}^{\{+, -\}})$ where $a' = \reserved@a \in \{a \cdot c, a\}$. Then if $e < 9000$ pt being a finite extension, we let $z \leftarrow z \pm e$ according to s , i.e. + if $s = \emptyset$ while - if $s = '-'$. Otherwise, i.e., $e \geq 9000$ pt for a infinite extension, let e' be the value shown below by invoking `\pcol@bg@ext@inf@d`

$$e' = \begin{cases} -(W_M - W_R) & d = l \\ W_P - (W_M - W_R) & d = r \\ -(H_M - H_R) & d = t \\ H_P - (H_M - H_R) & d = b \end{cases}$$

where $W_M - W_R$ is specified by `\pcol@bg@pageleft` and $H_M - H_R$ by `\pcol@bg@pagetop`. Then we let $z = e' \pm (e - 10000 \text{ pt})$ according to s again, i.e., move z *inside* by $(10000 \text{ pt} - e)$ from e' .

```

716 \def\pcol@bg@addext#1#2#3{%
717 \dimen@ii\@nameuse{pcol@bg@ext@#3@\reserved@a}\relax
718 \ifdim\dimen@ii<9000\p@\relax \advance#1#2\dimen@ii
719 \else
720 \pcol@bg@calculate#1\z@\@nameuse{pcol@bg@ext@inf@#3}}%
721 \advance\dimen@ii-\@M\p@ \advance#1#2\dimen@ii
722 \fi}
723 \def\pcol@bg@ext@inf@l{\pcol@bg@negative\pcol@bg@pageleft}
724 \def\pcol@bg@ext@inf@r{\pcol@bg@negative\pcol@bg@pageleft
725 \pcol@bg@paperwidth}
726 \def\pcol@bg@ext@inf@t{\pcol@bg@negative\pcol@bg@pagetop}
727 \def\pcol@bg@ext@inf@b{\pcol@bg@negative\pcol@bg@pagetop
728 \pcol@bg@paperheight}
729

```

`\pcol@bg@paperwidth` `\pcol@bg@paperheight` `\pcol@bg@pageleft` `\pcol@bg@pagetop` `\pcol@bg@textheight` `\pcol@bg@columnleft` `\pcol@bg@columnright` `\pcol@bg@columnwidth` `\pcol@bg@columnsep` The following macros specify the whole or a part of $F \in \{F_x, F_y, F_w, F_h\}$ being the body of `\pcol@bg@@a` for background painting regions $R_a^{[c]}$.

$$\pcol@bg@paperwidth = W_P - 2W_R = \paperwidth - 2\pagemargin \quad (t, T, b, B, r, R)$$

$$\pcol@bg@paperheight = H_P - 2H_R = \paperheight - 2\pagemargin \quad (b, B)$$

$$\pcol@bg@pageleft = W_M - W_R = (\pcol@bg@leftmargin + 1in) - \pagemargin \quad (t, T, b, B, l, L, r, R)$$

$$\pcol@bg@pagetop = H_M - W_R = (\topmargin + \headheight + \headsep + 1in) - \pagemargin \quad (t, T, b, B)$$

$$\pcol@bg@textheight = H_T = \textheight + \maxdepth \quad (b, B, l, L, r, R, C, G, n, N, p, P)$$

$$\pcol@bg@columnleft = W_c = \sum_{d=C_b^0}^{c-1} (w_c + g_c) \quad (c, C)$$

$$\pcol@bg@columnright = W_c + w_c = \pcol@bg@columnleft + \pcol@bg@columnwidth \quad (g, G)$$

$$\pcol@bg@columnwidth = w_c = \pcol@columnwidth \cdot c \quad (c, C)$$

$$\pcol@bg@columnsep = g_c = \pcol@columnsep \cdot c \quad (g, G)$$

Note that `\pagemargin` in F means $-\pagemargin$ because its sign is reversed by `\pcol@bg@paint@i`. The macros are used in `\pcol@bg@@a` whose region identifier a is shown in parentheses above, but besides them `\pcol@bg@paperwidth` is also used in `\pcol@bg@ext@inf@r`, `\pcol@bg@paperheight` in `\pcol@bg@ext@inf@b`, `\pcol@bg@pageleft` in `\pcol@bg@ext@inf@l` and `\pcol@bg@ext@inf@r`, and `\pcol@bg@pagetop` in `\pcol@bg@ext@inf@t` and `\pcol@bg@ext@inf@b`. Also note that `\pcol@bg@textheight` is used in `\pcol@output@clear` while it is temporarily redefined in `\pcol@output@start` and `\pcol@output@end`.

```

730 \def\pcol@bg@paperwidth{\@elt\paperwidth \@elt{2\pagemargin}}
731 \def\pcol@bg@paperheight{\@elt\paperheight \@elt{2\pagemargin}}
732 \def\pcol@bg@pageleft{\@elt{1in}\@elt\pcol@bg@leftmargin \@elt\pagemargin}
733 \def\pcol@bg@pagetop{\@elt{1in}\@elt\topmargin \@elt\headheight \@elt\headsep
734 \@elt\pagemargin}
735 \def\pcol@bg@textheight{\@elt\textheight \@elt\maxdepth}
736 \def\pcol@bg@columnleft{%

```

```

737 \@tempcnta\pcol@bg@from \@whilenum\@tempcnta<\pcol@currcol\do{%
738   \@elt{\@nameuse{pcol@columnwidth\@tempcnta}}%
739   \@elt{\@nameuse{pcol@columnsep\@tempcnta}}%
740   \advance\@tempcnta\@ne}}
741 \def\pcol@bg@columnright{\pcol@bg@columnleft \pcol@bg@columnwidth}
742 \def\pcol@bg@columnwidth{\@elt{\@nameuse{pcol@columnwidth\@tempcnta}}}
743 \def\pcol@bg@columnsep{\@elt{\@nameuse{pcol@columnsep\@tempcnta}}}

```

Besides the macros shown above, `\pcol@bg@a` uses the following macros defined by macros using `\pcol@bg@paintpage`, `\pcol@bg@paintcolumns` or `\pcol@bg@paintbox`.

- `\pcol@bg@preposttop` being `\pcol@bg@preposttop@left` or `\pcol@bg@preposttop@right` for $a \in \{p, P\}$ by `\@outputpage` and `\pcol@output@end`, the latter of which may define only the left one if the closing environment is not parallel-paged. That is, both of left and right macros are usually equivalent, but the right one can be smaller than the left if we have two or more (last pages of) `paracol` environments in a page and the closing environment is not parallel-paged while some others are. In such case, `\@outputpage` or `\pcol@output@start`, another macro referring to them, must paint the region below `\pcol@bg@preposttop@right` in the right page as a part of pre-environment stuff or post-environment stuff by `\letting \pcol@bg@preposttop` be `\pcol@bg@preposttop@left` and `\pcol@bg@preposttop@right` for the left and right parallel-pages respectively. Both macros have common initial value 0.
- `\pcol@bg@column@top` and `\pcol@bg@column@height` for $a \in \{c, g\}$ by `\pcol@buildcol@seprule` and `\pcol@build@cselt`.
- `\pcol@bg@spanning@top` and `\pcol@bg@spanning@height` for $a \in \{s, S\}$ by `\pcol@build@cselt`.
- `\pcol@bg@float@height` for $a \in \{f, F\}$ by `\pcol@output@elt`, `\pcol@ioutput@elt`, `\pcol@output@clear` and `\pcol@make@flushed@page`
- `\pcol@bg@footnote@height` for $a \in \{n, N\}$ by `\pcol@ioutput@elt`, `\pcol@i@make@flushed@page` and `\pcol@output@end`.

```

744 \def\pcol@bg@preposttop@left{0pt}
745 \let\pcol@bg@preposttop@right\pcol@bg@preposttop@left
746

```

The macros `\pcol@bg@a` define arguments F_x , F_y , F_w and F_h to be passed to `\pcol@bg@paintregion@i` in their bodies to calculate (x_0, y_0) and $(x_1 - x_0, y_1 - y_0)$ for regions $R_a^{[c]}$ as shown below in the form of $(x_0, y_0) + (x_1 - x_0, y_1 - y_0)$ (to have (x_1, y_1)), where $H^c = \text{\pcol@bg@columnntop}$, $h^c = \text{\pcol@bg@columnheight}$, $H^s = \text{\pcol@bg@spanningtop}$, $h^s = \text{\pcol@bg@spanningheight}$, $h^f = \text{\pcol@bg@floatheight}$, $h^n = \text{\pcol@bg@footnoteheight}$ and $H^p = \text{\pcol@bg@preposttop}$ calculated by macros which invoke background painting macros, while $s^n = \text{\skip\footins}$.

$R_c^c : (W_c, H^c) + (w_c, h^c)$
 $R_C^c : (W_c, 0) + (w_c, H_T)$
 $R_g^c : ((W_c + w_c), H^c) + (g_c, h^c)$
 $R_G^c : ((W_c + w_c), 0) + (g_c, H_T)$
 $R_{\{s,S\}} : (0, H^s) + (W_T, h^s)$
 $R_{\{t,T\}} : (-(W_M - W_R), -(H_M - H_R)) + ((W_P - 2W_R), H_M - H_R)$
 $R_{\{b,B\}} : (-(W_M - W_R), H_T) + ((W_P - 2W_R), (H_P - 2H_R) - ((H_M - H_R) + H_T))$
 $R_{\{l,L\}} : (-(W_M - W_R), 0) + ((W_M - W_R), H_T)$
 $R_{\{r,R\}} : (W_T, 0) + ((W_P - 2W_R) - ((W_M - W_R) + W_T), H_T)$
 $R_{\{f,F\}} : (0, 0) + (W_T, h^f)$
 $R_{\{n,N\}} : (0, H_T - (h^n + s^n)) + (W_T, h^n + s^n)$
 $R_{\{p,P\}} : (0, H^p) + (W_T, H_T - H^p)$

```

747 \def\pcol@bg@c{%
748   {\pcol@bg@columnleft}%
749   {\@elt\pcol@bg@columnntop}%
750   {\pcol@bg@columnwidth}%
751   {\pcol@bg@columnheight}}
752 \def\pcol@bg@C{%
753   {\pcol@bg@columnleft}%
754   {}}%
755   {\pcol@bg@columnwidth}%
756   {\pcol@bg@textheight}}
757 \def\pcol@bg@g{%
758   {\pcol@bg@columnright}%
759   {\@elt\pcol@bg@columnntop}%
760   {\pcol@bg@columnsep}%
761   {\pcol@bg@columnheight}}
762 \def\pcol@bg@G{%
763   {\pcol@bg@columnright}%
764   {}}%
765   {\pcol@bg@columnsep}%
766   {\pcol@bg@textheight}}
767 \def\pcol@bg@s{%
768   {}}%
769   {\pcol@bg@spanningtop}%
770   {\@elt\textwidth}%
771   {\pcol@bg@spanningheight}}
772 \def\pcol@bg@t{%
773   {\pcol@bg@negative\pcol@bg@pageleft}%
774   {\pcol@bg@negative\pcol@bg@pagetop}%
775   {\pcol@bg@paperwidth}}%

```

```

776 {\pcol@bg@pagetop}}
777 \def\pcol@bg@b{%
778 {\pcol@bg@negative\pcol@bg@pageleft}%
779 {\pcol@bg@textheight}%
780 {\pcol@bg@paperwidth}%
781 {\pcol@bg@paperheight
782 \pcol@bg@negative{\pcol@bg@pagetop \pcol@bg@textheight}}}
783 \def\pcol@bg@l{%
784 {\pcol@bg@negative\pcol@bg@pageleft}%
785 {}%
786 {\pcol@bg@pageleft}%
787 {\pcol@bg@textheight}}
788 \def\pcol@bg@r{%
789 {\@elt\textwidth}%
790 {}%
791 {\pcol@bg@paperwidth
792 \pcol@bg@negative{\pcol@bg@pageleft \@elt\textwidth}}}
793 {\pcol@bg@textheight}}
794 \def\pcol@bg@f{%
795 {}%
796 {}%
797 {\@elt\textwidth}%
798 {\pcol@bg@floatheight}}
799 \def\pcol@bg@n{%
800 {}%
801 {\pcol@bg@textheight
802 \pcol@bg@negative{\pcol@bg@footnoteheight \@elt{\skip\footins}}}%
803 {\@elt\textwidth}%
804 {\pcol@bg@footnoteheight \@elt{\skip\footins}}}
805 \def\pcol@bg@p{%
806 {}%
807 {\@elt\pcol@bg@preposttop}%
808 {\@elt\textwidth}%
809 {\pcol@bg@textheight \pcol@bg@negative{\@elt\pcol@bg@preposttop}}}
810 \let\pcol@bg@S\pcol@bg@s
811 \let\pcol@bg@T\pcol@bg@t
812 \let\pcol@bg@B\pcol@bg@b
813 \let\pcol@bg@L\pcol@bg@l
814 \let\pcol@bg@R\pcol@bg@r
815 \let\pcol@bg@F\pcol@bg@f
816 \let\pcol@bg@N\pcol@bg@n
817 \let\pcol@bg@P\pcol@bg@p
818

```

11 Special Output Routines

11.1 Dispatcher

`\pcol@op@start` The macro `\pcol@op@.f` where $f \in F = \{\text{start, switch, flush, clear, end}\}$ has our own
`\pcol@op@switch` `\outputpenalty` code less than -10000 to invoke the corresponding macro `\pcol@output@.f`.
`\pcol@op@flush` The code macros are given to `\pcol@invokeoutput` as its argument by `\pcol@zparacol`
`\pcol@op@clear` ($f = \text{start}$), `\pcol@switchcol` ($f = \text{switch}$), `\pcol@visitallcols` ($f = \text{switch}$), `\pcol@`
`\pcol@op@end` `com@flushpage` ($f = \text{flush}$), `\pcol@com@clearpage` ($f = \text{clear}$), `\pcol@flushclear` ($f =$

switch), and `\endparacol` ($f = \text{end}$) to set one of them into `\outputpenalty`, so that the other user `\pcol@specialoutput` examines which special function is invoked.

```
819 %% Special Output Routines: Dispatcher
820
821 \def\pcol@op@start{-10010}
822 \def\pcol@op@switch{-10011}
823 \def\pcol@op@flush{-10012}
824 \def\pcol@op@clear{-10013}
825 \def\pcol@op@end{-10014}
826
```

`\pcol@specialoutput` The macro `\pcol@specialoutput` is invoked solely in `\pcol@output` to invoke our own or L^AT_EX's special output routine. It examines if $P = \text{\outputpenalty} \in \{\text{\pcol@op@.}f \mid f \in F\}$ and then, if so, before invoking `\pcol@output@.f`, we rebuild `\@holdpg` removing `\lastbox` and the last vertical skip as done in L^AT_EX's `\@specialoutput`. We also let `\outputpenalty = -10000`¹⁶⁴ so that `\vsize` is correctly set to `\@colroom` in the second half of `\pcol@output` after this macro finishes.

Otherwise, i.e., if $P \notin \{\text{\pcol@op@.}f \mid f \in F\}$, we simply invokes L^AT_EX's `\@specialoutput`¹⁶⁵.

```
827 \def\pcol@specialoutput{%
828   \ifnum\outputpenalty=\pcol@op@start\relax
829     \let\reserved@a\pcol@output@start
830   \else\ifnum\outputpenalty=\pcol@op@switch\relax
831     \let\reserved@a\pcol@output@switch
832   \else\ifnum\outputpenalty=\pcol@op@flush\relax
833     \let\reserved@a\pcol@output@flush
834   \else\ifnum\outputpenalty=\pcol@op@clear\relax
835     \let\reserved@a\pcol@output@clear
836   \else\ifnum\outputpenalty=\pcol@op@end\relax
837     \let\reserved@a\pcol@output@end
838   \else \let\reserved@a\@specialoutput
839   \fi\fi\fi\fi\fi
840   \ifnum\outputpenalty=-\@Miv\relax
841     \ifvoid\footins\else \pcol@Log\dummy{dummy}\footins \fi
842   \fi
843   \ifx\reserved@a\@specialoutput\else
844     \global\setbox\@holdpg\vbox{\unvbox\@holdpg \unvbox\@cclv
845       \setbox\@tempboxa\lastbox \unskip}%
846     \outputpenalty-\@M
847   \fi
848   \reserved@a}
849
```

11.2 Building Starting Page

`\pcol@output@start` The macro `\pcol@output@start` is invoked solely from `\pcol@specialoutput` to process the special `\output` request made in `\pcol@zparacol` and to build the *starting page* from which parallel columns start possibly with the stuff preceding `\begin{paracol}`, or *pre-environment stuff* in short. First, we turn `\ifpcol@output = true` so that `\output` requests for page breaks are processed by our own macros such as `\pcol@makecol` hereafter. Then we let $p = p_b =$

¹⁶⁴It can be any value larger than -10004.

¹⁶⁵With footnote logging if `\outputpenalty = -10004`.

$p_t = 0$ and $\Pi = \emptyset$ because we have nothing for $q < p_t = 0$. We also move `\@deferlist` to `\@dbldeferlist` and then let `\@deferlist` be empty because all column-wise deferred floats become page-wise. In this float importation, as discussed in item-(4) of §1.8, we force all floats in the list have depth 0 to ensure no one has `1sp` to conform our own and old-fashioned page-wise float placement mechanism¹⁶⁶. We then and let $\Phi = \perp$ because we don't have any deferred footnotes.

Next we calculate $H = H_r - (H_m + H_f + H_b)$ where $H_r = \text{\@colroom}$; H_m is the height-plus-depth of the main vertical list in `\@holdpg`; H_f is the sum of `\skip\footins`, the height-plus-depth of `\footins` and `\belowfootnoteskip`, if `\footins` is not \perp or 0 otherwise; and $H_b = \text{\textfloatsep}$ if the pre-environment stuff has bottom floats or 0 otherwise. That is, H is the room for each of column-page in the starting page. Then we examine if $H < 1.5 \times \text{\baselineskip}$ to mean `\pcol@output` would force a page break with warning. If so, we assume we have a page break before `\begin{paracol}` to ship out pre-environment stuff to avoid the warning. Therefore, we invoke L^AT_EX's `\@makecol`¹⁶⁷ giving it `\@holdpg` through `\@cclv` to build the ship-out image in `\@outputbox`. Then the box is passed to `\@outputbox` for which we temporarily let `\ifpcol@output = false` because the page is assumed to be outside the `paracol` environment having just started.

After that we invoke `\pcol@startpage` to let it produce $\pi(p_t)$ for the starting page $p_t = 0$ letting `\pcol@currpage` be empty so that the macro will not refer to it. The page $\pi(0)$ is usually empty but can have non-empty $\pi^i(0)$ with imported deferred floats which are now page-wise. Moreover, we can have two or more pages if deferred page-wise floats produce float pages. However, we can be unaware of these effects of floats because the resulting Π^+ with them is correct of course.

Then let `\topskip = \pcol@topskip` being the value at `\begin{paracol}`, and `\ifpcol@firstpage = false`, because we have the starting page without pre-environment stuff and thus the first item of each column will be at its top.

```

850 %% Special Output Routines: Building First Page
851
852 \def\pcol@output@start{%
853   \global\pcol@outputtrue
854   \global\pcol@page\z@ \global\pcol@toppage\z@ \global\pcol@basepage\z@
855   \global\let\pcol@pages\empty
856   \global\let\@dbldeferlist\@deferlist \global\let\@deferlist\empty
857   {\def\@elt##1{\global\dp##1\z@}\@dbldeferlist}%
858   \setbox\z@\box\pcol@topfnotes
859   \@tempdima\@colroom
860   \advance\@tempdima-\ht\@holdpg \advance\@tempdima-\dp\@holdpg
861   \ifvoid\footins\else
862     \advance\@tempdima-\skip\footins
863     \advance\@tempdima-\ht\footins \advance\@tempdima-\dp\footins
864     \advance\@tempdima-\belowfootnoteskip
865   \fi
866   \ifx\@botlist\empty\else \advance\@tempdima-\textfloatsep \fi
867   \ifdim\@tempdima<1.5\baselineskip
868     \setbox\@cclv\box\@holdpg \@makecol
869     \pcol@outputfalse \@outputpage \pcol@outputtrue
870     \global\let\pcol@currpage\empty \pcol@startpage
871     \global\topskip\pcol@topskip \global\pcol@firstpagefalse

```

¹⁶⁶Though having `1sp` is almost impossible.

¹⁶⁷We can be unaware of our customization for synchronization in `\pcol@combinefloats` because `\pcol@textfloatsep` is made ∞ by `\pcol@zparacol`.

Otherwise, i.e., if $H \geq 1.5 \times \text{\baselineskip}$, we invoke `\pcol@makenormalcol` to make the pre-environment stuff as the spanning stuff of the starting page. The macro is different from `\@makecol` as follows; the height of resulting `\@outputbox` is natural rather than `\textheight`; merged footnotes is excluded if any; and the skip of `\textfloatsep` is added below the bottom floats also if any,

Then we let h be the height-plus-depth of `\@outputbox` being the spanning stuff and shrink `\@colht` by h . Next if $h > H_B = \text{\pcol@bg@preposttop} \in \{\text{\pcol@bg@preposttop@left}, \text{\pcol@bg@preposttop@right}\}$, being the bottom of the previous `paracol` environment (having right parallel-page) or 0 if the current page does not have it, to mean we have ordinary single-columned stuff in pre-environment stuff, we paint its background by `\pcol@bg@paintbox` temporarily letting `\pcol@bg@textheight = h` so that $y_0 = H_B$ and $y_1 = h$ for $R_{\{p,P\}}$. This background painting is not only for $\pi^b(0)$ which we acquire from from `\@freelist` by `\@next` and let have the spanning stuff, but also for `\pcol@rightpage` if $C_L < C$ to mean parallel-paging for which we temporarily increment `\c@page` by one if non-paired.

We also let $\pi^h(0)$ be the shrunk `\@colht`, and $\pi^t(0)$ be `\topskip` if $h = 0$ assuming that the page does not have any spanning stuff¹⁶⁸ to typeset column-pages from the top of the page, or otherwise be 0 together with `\topskip` to inhibit the ordinary `\topskip` insertion.

As for $\pi^m(0)$, we define it as follows, referring to $\mathcal{M} = \text{\pcol@mparbottom@out} = \{M_L^l, M_L^r, M_R^l, M_R^r\}$, where M_X^x has exactly one element $\text{mpar}(h, t)$ which may be the position of last marginal notes in the last `paracol` environment in the page we are working on, or $M_X^x = \{\text{mpar}(0, 0)\}$ if such marginal note or the environment itself does not exist in the page. On the other hand, $B = \text{\@mparbottom}$ may have non-zero for the bottom edge of the last marginal note in pre-environment stuff including the last `paracol` environment if any. Therefore, what we need to do is to let $M_L^x = \{\text{mpar}(0, B)\}$ to reflect the marginal node whose bottom is at B and which can be different from what M_L^x had, where x is the target margin in the pre-environment stuff determined by `\ifmparswitch`, the parity of $\text{page}(0)$ and `\ifreversemargin`.

The replacement is done by `\pcol@do@mpbout` which invokes `\pcol@do@mpbout@whole` (m_L^l) $\langle m_L^r \rangle \langle M_R^l \rangle \langle M_R^r \rangle$ where $m_L^x \in \{M_L^x, \text{\pcol@do@mpbout@elem}\langle M_L^x \rangle\}$ whose choice is made according that $x \in \{l, r\}$ is the target margin (latter) or not (former). Therefore, prior to the invocation of `\pcol@do@mpbout`, we `\define \pcol@do@mpbout@whole` so that it `\xdefines` $\mathcal{M} = \text{\pcol@mparbottom@out}$ with its four arguments, and `\pcol@do@mpbout@elem` to let it be expanded to `\@elt{0}{B} = \text{mpar}(0, B)`. After that, we also invoke `\pcol@bias@mpbout` giving it $-h$ to replace $\text{mpar}(t, b)$ being the sole element of each $M_{\{L,R\}}^{\{l,r\}}$ in the resulting \mathcal{M} with $\text{mpar}(t - h, b - h)$ to have what we give to $\pi^m(0)$. This replacement transforms the coordinates for text area to that for columns, and makes it possible for the first marginal note in each margin in the `paracol` environment we now start to exploit the space for pre-environment stuff even if it is tall extraordinarily.

Then we let $\pi(0)$ have $\pi^i(0)$ and $\pi^m(0)$ shown above, and $\pi^p(0) = \text{\c@page}$, $\pi^f(0) = \perp$ and $\pi^s(0) = \emptyset$ by `\pcol@defcurrpage`, and let `\ifpcol@firstpage = true` because $\pi^b(0)$ has pre-environment stuff.

```

872 \else
873   \pcol@makenormalcol
874   \@tempdima\ht\@outputbox \advance\@tempdima\dp\@outputbox
875   \global\advance\@colht-\@tempdima
876   \def\reserved@a{\%
877     \ifdim\pcol@bg@preposttop=\@tempdima\else

```

¹⁶⁸Checking the emptiness by `\pcol@ifempty` does not work well for the very first page of a document because it has a `\write` as the very first item.


```

878     \edef\pcol@bg@textheight{\@elt{\number\@tempdima sp}}%
879     \pcol@bg@paintbox{Pp}%
880     \fi}
881 \ifnum\pcol@ncolleft<\pcol@ncol
882   \global\setbox\pcol@rightpage\vbox{%
883     \ifpcol@paired\else \advance\c@page\@ne \fi
884     \let\pcol@bg@preposttop\pcol@bg@preposttop@right
885     \reserved@a \unvbox\pcol@rightpage}%
886 \fi
887 \pcol@Fb
888 \@next\@currbox\@freelist{\global\setbox\@currbox\vbox{%
889   \let\pcol@bg@preposttop\pcol@bg@preposttop@left
890   \reserved@a \unvbox\@outputbox}}\pcol@ovf
891 \pcol@Fe{output@start(preenv)}%
892 \global\dimen\@currbox\@colht
893 \ifdim\@tempdima=z@ \@tempskipa\topskip \else \@tempskipa z@ \fi
894 \global\skip\@currbox\@tempskipa \global\topskip\@tempskipa
895 \def\pcol@do@mpbout@whole##1##2##3##4{%
896   \xdef\pcol@mparbottom@out{##1}{##2}{##3}{##4}}%
897 \def\pcol@do@mpbout@elem\@elt##1##2{\@elt{0}{\number\@mparbottom}}%
898 \pcol@do@mpbout
899 \pcol@bias@mpbout{-\@tempdima}%
900 \pcol@defcurrpage{\number\c@page}\@currbox\voidb@x{\pcol@mparbottom@out}%
901 \global\pcol@firstpagetrue
902 \fi

```

Then regardless of H , we do the followings for all columns $c \in [0, C)$ to build κ_c , after initializing $\backslash\@colroom$ to be $\backslash\@colht$, and invoking $\backslash\@floatplacement$ to reinitialize the parameters of column-wise float placement.

First, if we have let $\backslash\@topskip = 0$ with the pre-environment stuff, we let $\kappa_c(\beta^b)$ have an invisible $\backslash\hrule$ whose height and depth are 0 as the very first vertical item of the column-page. When we visit the column c for the first time afterward, we will $\backslash\unvbox$ the box to let $\text{T}_{\text{E}}\text{X}$'s page builder have $\backslash\@topskip = 0$ and the invisible rule. Then the first vertical item of the column-page is added but it is recognized as non-first by $\text{T}_{\text{E}}\text{X}$'s page builder and thus it inserts $\backslash\@baselineskip$ referring to $\backslash\@prevdepth$ as the depth of the last item. The important issue is that the $\backslash\@prevdepth$ to be referred is assured having its value at $\backslash\@begin\{paracol\}$, which is usually the depth of the last item of spanning stuff, by the following mechanism: (1) $\backslash\@pcol@invokeoutput$ invoked in $\backslash\@pcol@zparacol$ saves $\backslash\@prevdepth$ in $\backslash\@pcol@prevdepth$ before the $\backslash\@output$ request for $\backslash\@pcol@output@start$; (2) $\backslash\@pcol@prevdepth$ is saved in $\kappa_c(\delta)$ by $\backslash\@pcol@setcurrcolnf$ invoked from $\backslash\@pcol@output@start$ as discussed afterward; (3) when the column c is visited for the first time, the special output routine $\backslash\@pcol@output@start$ itself ($c = 0$) or $\backslash\@pcol@output@switch$ ($c > 0$) restores $\backslash\@pcol@prevdepth$ from $\kappa_c(\delta)$ by $\backslash\@pcol@getcurrcol$; (4) $\backslash\@pcol@invokeoutput$ which made the $\backslash\@output$ request for (3) lets $\backslash\@prevdepth$ have the value of $\backslash\@pcol@prevdepth$ after the request. Therefore, the baseline progress from the last line of the spanning stuff to the first line of each column-page should be very natural as we see in the third and fourth lines of §6 of Part I.

Then we invoke $\backslash\@pcol@setcurrcolnf$ to save the following values for $\kappa_c(e)$ ($e \neq \beta$); $\kappa_c(\tau) = \backslash\@voidb@x$ because c does not have column-wise footnotes so far; $\kappa_c(\delta) = \backslash\@prevdepth$ as discussed above; $\kappa_c(\lambda_t) = \kappa_c(\lambda_m) = \kappa_c(\lambda_b) = \emptyset$ because $\backslash\@pcol@makenormalcol$ and $\backslash\@combinefloats$ invoked from it emptied them; $\kappa_c(\lambda_d) = \emptyset$ as discussed above; $\kappa_c(\nu_t) = \backslash\@c@topnumber$, $\kappa_c(\nu_b) = \backslash\@c@botnumber$ and $\kappa_c(\nu_t) = \backslash\@c@totalnumber$ as initialized by $\backslash\@floatplacement$ invoked from $\backslash\@floatplacement$; $\kappa_c(\rho_t) = \backslash\@topfraction \times \backslash\@colht$

and $\kappa_c(\rho_b) = \text{\bottomfraction} \times \text{\@colht}$ as initialized by `\@floatplacement`; $\kappa_c(\sigma)$ is defined by `\if@nobreak` and `\if@afterindent` at the time of `\begin{paracol}`; and $\kappa_c(\varepsilon) = \text{\everypar}$ at the time of `\begin{paracol}`. We also let $\kappa_c(\beta^p) = 0$ because $p = 0$ and $\kappa_c(\beta^r) = \text{\@colroom}$ defined above. In addition, we let $S_c = \emptyset$ because we don't have any column-page having been completed.

We also examine if $\hat{\gamma}_0^c = \text{\pcol@columncolor} \cdot c$ is defined and, if so, acquire an `\insert` from `\@freelist` to let $\gamma_0^c = \text{\pcol@columncolor@box} \cdot c$ have the coloring `\special` for the color defined in $\hat{\gamma}_0^c$ by invoking `\pcol@set@color` being the original `\set@color` with nullification of `\aftergroup`. Otherwise, we let $\gamma_0^c = \perp$.

```

903 \global\@colroom\@colht \pcol@floatplacement
904 \pcol@currcol\z@ \@whilenum\pcol@currcol<\pcol@ncol\do{%
905   \pcol@Fb
906   \@next\@currbox\@freelist{\global\setbox\@currbox\vbox{%
907     \ifdim\topskip=\z@ \hrule\@height\z@\@width\z@ \fi}}\pcol@ovf
908   \pcol@Fe{output@start(col)}%
909   \pcol@setcurrcolnf
910   \global\count\@currbox\z@
911   \global\dimen\@currbox\@colroom
912   \expandafter\gdef\csname pcol@shipped\number\pcol@currcol\endcsname{}%
913   \pcol@ifccdefined
914     {\@next\@currbox\@freelist{\global\setbox\@currbox\vbox{%
915       \def\current@color{\pcol@ccuse{}}\let\aftergroup\@gobble
916       \pcol@set@color}}\pcol@ovf}%
917     {\def\@currbox{\voidb@x}}%
918   \pcol@ccxdef{\@currbox}%
919   \advance\pcol@currcol\@ne}%

```

Finally, we let $c = \text{\pcol@currcol} = 0$ for the first column, and regain the parameters in κ_0 by `\pcol@getcurrcol`. Then before putting $\kappa_0(\beta^b)$ to the main vertical list by `\unvbox` returning $\kappa_0(\beta)$ to `\@freelist` by `\@cons` because it has become useless so far, we save the color context just with γ_0^c into Γ_s by `\pcol@savecolorstack` because Γ is emptied by `\pcol@zparacol`. Then we `\insert \footins` through itself to the main vertical list if it is not \perp and thus has footnotes to be merged. This `\insertion` is different from other footnote `\insertion` because `\footins` is not `\unvboxed` but is put as a whole and is followed by `\penalty\interlinepenalty`, so that footnotes will not be broken by T_EX's page builder to prevent the reconnection of a broken footnote with inappropriate glue discarding, which we will discuss in §18. We also add a penalty 10000 or `\interlinepenalty` according to `\if@nobreak = true` or not to allow the first column to start from a new page when its first item is taller than the room in the starting page.

```

920 \global\pcol@currcol\z@
921 \pcol@getcurrcol
922 \pcol@savecolorstack
923 \pcol@Fb
924 \@cons\@freelist\@currbox \unvbox\@currbox
925 \pcol@Fe{output@start(col)}%
926 \ifvoid\footins\else
927   \pcol@Log\pcol@output@start{insert}\footins
928   \insert\footins{\box\footins\penalty\interlinepenalty}%
929 \fi
930 \if@nobreak \nobreak \else \addpenalty\interlinepenalty \fi
931

```

`\pcol@makenormalcol` The macro `\pcol@makenormalcol` is invoked solely from `\pcol@output@start` to let `\@outputbox` have the pre-environment stuff as spanning stuff of the starting page. The operations this macro performs are very similar to those of `\@makecol`, which in fact is used in this macro itself, but has the following differences.

- (1) If `\ifpcol@mgfnote = true`, we exclude footnotes in `\footins` from `\@outputbox`, because they are merged with page-wise footnotes given in columns, by saving it into `\@tempboxa` during the building process and restoring it from the box register.
- (2) If pre-environment stuff does not have bottom floats, we build `\@outputbox` by ourselves without relying on `\@makecol` because the skips put into the bottom (or near it) by the macro is harmful to making pre-environment stuff and parallel columns naturally connected. Therefore, we move `\@holdpg` to `\@outputbox` adding `\footins` to its tail if any by `\pcol@combinefootins`, and then combine top floats if any by `\pcol@combinefloats`¹⁶⁹. In addition, we clear `\@midlist` and returns its contents to `\@freelist` as `\@makecol` does.
- (3) If pre-environment stuff has bottom floats, on the other hand, we use `\@makecol` to build pre-environment stuff in `\@outputbox` moving `\@holdpg` into `\box255` prior to the invocation¹⁷⁰. Also before invocation in addition, we temporarily let `\ifpcol@lastpage = true` to let `\@combinefloats = \pcol@combinefloats` used in `\@makecol` put a vertical skip of `\textfloatsep` below the bottom floats so that the floats are well separated from the top of multi-column stuff in the starting page. We also nullify `\@textbottom` by making it `\let@equal to \relax` because it is unnecessary to put an infinitely stretchable skip at the bottom¹⁷¹, and let `\vbadness = 10000` to avoid an inevitable underfull message because `\@makecol` lets `\@outputbox` as tall as `\textheight`.
- (4) In both cases but especially that with bottom floats, resulting `\@outputbox` is decapsulated by `\unvbox` to make its height *natural*.

Note that the special function for synchronized column-page in `\pcol@combinefloats` used directly or indirectly in this macro, on the other hand, is not active in the invocation because `\pcol@zparacol` initialized `\pcol@textfloatsep = ∞` to mean we have no synchronization points. Also note that bottom floats and non-merged footnotes are put in `\@outputbox` and thus they will not appear at the bottom of the page but above the column-pages in the page¹⁷².

```

932 \def\pcol@makenormalcol{%
933   \ifpcol@mgfnote \setbox\@tempboxa\box\footins \fi
934   \begingroup
935   \ifx\@botlist\@empty
936     \ifvoid\footins \setbox\@outputbox\box\@holdpg
937     \else          \pcol@combinefootins\@holdpg\footins
938     \fi
939     \pcol@Fb
940     \let\@elt\relax \xdef\@freelist{\@freelist\@midlist}%
941     \pcol@Fe{makenormalcol}%

```

¹⁶⁹Since we do not have bottom floats, the order of materials in the resulting `\@outputbox` being top floats, main text and footnotes should be consistent with other pages with any \LaTeX including $\text{p}\LaTeX$.

¹⁷⁰Therefore the order of footnotes and bottom floats is consistent with other pages and columns, i.e., footnote-first in the native \LaTeX while float-first in $\text{p}\LaTeX$, for example.

¹⁷¹Even if unharmed.

¹⁷²We could put them at the bottom by keeping them somewhere and insert them in `\pcol@outputcolumns`, but it will cause another problem that the numbers of the figures and footnotes are smaller than those in column-pages which are above them.

```

942 \global\let\@midlist\@empty
943 \pcol@combinefloats
944 \else
945 \pcol@lastpagetrue
946 \setbox\@cc1v\box\@holdpg \let\@textbottom\relax \vbadness\@M
947 \@makecol
948 \fi
949 \global\setbox\@outputbox\vbox{\unvbox\@outputbox}%
950 \endgroup
951 \ifpcol@mgfnote \setbox\footins\box\@tempboxa \fi}
952

```

11.3 Column-Switching

`\pcol@output@switch` The macro `\pcol@output@switch` is invoked from `\pcol@specialoutput` to process the special `\output` request made in `\pcol@switchcol`, `\pcol@visitallcols` and `\pcol@flushclear`, for a column-switching from $c = \text{\pcol@currcol}$ to $d = \text{\pcol@nextcol}$ which can be c . The macro is also invoked from `\pcol@makeflushedpage` to synchronize and to flush all current column-pages but staying in c .

First, we examine if the column-switching is to close a spanning text, i.e., `\ifpcol@sptext = true` and $c = 0$, and if so we do the following; let h_p be the height of `\pcol@prespan` having pre-spanning-text stuff if it is not \perp , or 0 if \perp to mean we have had a page break in the spanning text; add h_p to `\@colroom` which we temporarily shrank when the spanning text starts; add an element $\text{span}(H, h)$ to the tail of $\pi^s(p_t) = \text{\pcol@sptextlist}$ by `\pcol@getcurrpage` and `\pcol@defcurrpage`, where H is h_p plus the total height of top floats measured by `\pcol@addflhd`, and h is the height-plus-depth of `\@holdpg` having (a part of) spanning text, represented in the form of integers and thus expanded with `\number`; shift `\@holdpg` left by `\pcol@shiftspanning` if the column-0 is not the leftmost due to column-swapping; and then put pre-spanning-text stuff and (maybe shifted) `\@holdpg` into `\@holdpg` itself so as to let `\@holdpg` have everything in the column-page 0 as usual.

Note that it can be `\pcol@prespan = \perp` if spanning text had a page break (or multiple ones) in it as shown above. This empty pre-spanning-text stuff, however, does not always mean that we have no top floats because the page break in the spanning text can produce a column-page with top floats which are deferred from the previous page(s), or though unlikely the spanning text itself has float environments. Therefore, the measurement of the total height of top floats are always necessary. Also note that we perform these operations at the first column-switching for column-scanning from $c = 0$ with `\ifpcol@sptext = true`, i.e., prior to the synchronization itself which takes place afterward, as explained shortly.

Then regardless of the operations above, we acquire an `\insert` from `\@freelist` by `\@next` for $\kappa_c(\beta)$ to store the current column-page in (maybe modified) `\@holdpg` by `\pcol@clearcst@unvbox` to add uncoloring `\specials` to rewind the color stack Γ^c at the bottom and possibly coloring ones to establish that saved in Γ_s at the top as the color context for the column-page when it has the first item.

Then if `\footins $\neq \perp$` , we perform one of the followings.

- If page-wise footnote typesetting is in effect and $p = p_t$, we save `\footins` into $\pi^f(p)$ by the sequence of `\pcol@getcurrpinfo` to get $\pi(p)$, `\pcol@savefootins` to move it in $\pi^f(p)$, and `\pcol@defcurrpage` to update $\pi(p)$ with $\pi^f(p)$.
- If page-wise footnote typesetting is in effect but $p < p_t$, we simply discard the contents of `\footins` by making it \perp , because `\footins` should have $\pi^f(p)$ which has been already fixed.

- If column-wise footnote typesetting is in effect, by `\pcol@savefootins` we save `\footins` into `\pcol@currfoot`, which should be \perp in other cases, so that it will be saved into $\kappa_c(\tau)$ by `\pcol@setcurrcol` afterward.

Then if $c = 0$, we invoke `\pcol@setpageno` to reflect the jump of `\c@page` made in the building process of the column-page to $\pi(q)$ for all $q \in [p, p_t]$. After that, we save c 's column-context into κ_c by `\pcol@setcurrcol` and let $\kappa_c(\beta^p) = p$ and $\kappa_c(\beta^r) = \backslash@colroom$.

```

953 %% Special Output Routines: Column-Switching
954
955 \def\pcol@output@switch{%
956   \ifpcol@sptext\ifnum\pcol@currcol=\z@
957     \ifvoid\pcol@prespan \dimen@\z@ \else \dimen@\ht\pcol@prespan \fi
958     \global\advance\@colroom\dimen@
959     \pcol@addflhd\@toplist\pcol@textfloatsep
960     \pcol@getcurrpinfo\@tempcnta\@tempdima\@tempskipa
961     \@tempdimb\ht\@holdpg \advance\@tempdimb\dp\@holdpg
962     \@cons\pcol@sptextlist{\number\dimen@}{\number\@tempdimb}}%
963     \pcol@defcurrpage{\number\@tempcnta}\pcol@spanning\pcol@footins
964         {\pcol@sptextlist}{\pcol@mparbottom}%
965     \pcol@shiftspanning\@holdpg
966     \setbox\@holdpg\vbox{\unvbox\pcol@prespan \unvbox\@holdpg}%
967   \fi\fi
968   \pcol@Fb
969   \@next\@currbox\@freelist{\global\setbox\@currbox\vbox{
970     \pcol@clearcst\unvbox\@holdpg}}\pcol@ovf
971   \pcol@Fe{output@switch}%
972   \def\pcol@currfoot{\voidb@x}%
973   \ifvoid\footins\else
974     \ifpcol@scfnote
975       \ifnum\pcol@page=\pcol@toppage
976         \pcol@getcurrpinfo\@tempcnta\@tempdima\@tempskipa
977         \pcol@Log\pcol@output@switch{save}\footins
978         \pcol@Fb
979         \pcol@savefootins\pcol@footins
980         \pcol@Fe{output@switch(pagefn)}%
981         \pcol@defcurrpage{\number\@tempcnta}\pcol@spanning\pcol@footins
982             {\pcol@sptextlist}{\pcol@mparbottom}%
983       \else
984         \pcol@Log\pcol@output@switch{discard}\footins
985         \setbox\@tempboxa\box\footins
986         \fi
987     \else
988       \pcol@Log\pcol@output@switch{save}\footins
989       \pcol@Fb
990       \pcol@savefootins\pcol@currfoot
991       \pcol@Fe{output@switch(colfn)}%
992     \fi
993   \fi
994   \ifnum\pcol@currcol=\z@ \pcol@setpageno \fi
995   \pcol@setcurrcol
996   \global\count\@currbox\pcol@page
997   \global\dimen\@currbox\@colroom

```

Next, we examine if `\ifpcol@sptext = true` and $c = 0$ again, and if so we *broadcast* `\if@nobreak` and `\if@afterindent`, or in other words $\kappa_c(\sigma)$, and tokens in `\everypar = \kappa_c(\varepsilon)`,

to pretend all columns follow the spanning text. That is, for each column e , we restore its column-context from κ_e by `\pcol@getcurrcol`, let `\if@nbreak` and `\if@afterindent` have the values for $c \in \{0, C-1\}$ and `\everypar = $\kappa_c(\varepsilon)$` , and then save the context to κ_e by `\pcol@setcurrcol` so that $\kappa_e(\sigma) = \kappa_c(\sigma)$ and $\kappa_e(\varepsilon) = \kappa_c(\varepsilon)$. After that, we `\globally` turn `\ifpcol@sptext = false` to give it the default state.

Note that this broadcast is essential when the spanning text has sectioning commands to have consistent settings of the page break inhibition, the skip above the another sectioning command following them, and the indentation of the first paragraph, for all columns. On the other hand, broadcasting of `\everypar` is natural even when it does not have sectioning commands because all columns may be considered following the spanning text. Also note that, as mentioned in the explanation of the first examination at the beginning of this macro, we perform these operations at the first column-switching for column-scanning from $c = 0$ with `\ifpcol@sptext = true` prior to the synchronization following the spanning text. This means, if `\if@nbreak = true`, `\penalty = 10000` is inserted at the top and bottom end of the space for spanning text in the columns such that $c \neq 0$, the former by this column-scan and the latter by the column-switching to c made after the synchronization. Therefore, if our synchronization mechanism and \TeX 's page builder once agreed both end can be in a page, both end will not chosen as page break points¹⁷³.

```

998 \let\reserved@a@\nbreakfalse \let\reserved@b@\afterindentfalse
999 \ifpcol@sptext\ifnum\pcol@currcol=\z@
1000 \if@nbreak \let\reserved@a@\nbreaktrue \fi
1001 \if@afterindent \let\reserved@b@\afterindenttrue \fi
1002 \@temptokena\everypar
1003 \pcol@currcol\z@ \@whilenum\pcol@currcol<\pcol@ncol\do{%
1004   \pcol@getcurrcol \reserved@a \reserved@b \everypar\@temptokena
1005   \pcol@setcurrcol
1006   \advance\pcol@currcol\@ne}%
1007 \global\pcol@sptextfalse
1008 \fi\fi

```

Finally we invoke `\pcol@sync` for the synchronization if `\ifpcol@sync` or `\ifpcol@clear` is `true`, and then `\pcol@restartcolumn` to restart the current column-page d if `\ifpcol@clear = false` to mean ordinary (but possibly synchronized) column-switching or `\ifpcol@clear = true` but `\ifpcol@sync = true` too to mean pre-flushing column height check, before finishing `\output` routine letting `\ifpcol@sync = false` for next column-switching.

```

1009 \@tempswafalse \ifpcol@sync \@tempwattrue \fi \ifpcol@clear \@tempwattrue \fi
1010 \if@tempswa \pcol@sync \fi
1011 \@tempwattrue
1012 \ifpcol@clear \ifpcol@sync\else \@tempswafalse \fi\fi
1013 \if@tempswa \pcol@restartcolumn \fi
1014 \global\pcol@syncfalse}
1015

```

`\pcol@shiftspanning` The macro `\pcol@shiftspanning` is used in `\pcol@makecol` and `\pcol@output@switch` to let box register b have itself but shifted left by $W_T - w_0 = \text{\textwidth} - \text{\columnwidth}$ so that the left edge of its contents spanning text is aligned to the left edge of the leftmost column being different from column-0 due to column-swapping, i.e., if `\ifpcol@swapcolumn = true`

¹⁷³As for $c = 0$, its top end of spanning text is a feasible break point to make the penalty insertion asymmetric. Therefore, we need to reinvestigate if the condition of the broadcast is really appropriate, and, if inappropriate, have to go back to the old implementation in which `\ifpcol@sync` is included in the condition. Otherwise, if proved appropriate, we will have to consider to make the penalty insertion symmetric by adding `\nbreak` at the top of spanning text in $c = 0$.

and $\backslash\text{c@page} \bmod 2 = 0$. Note that $\backslash\text{c@page}$ is *not* obtained from $\pi^p(p)$ by the invokers but have the value when the $\backslash\text{output}$ request is made to let invokers work, and thus have the correct value even when a jump occurs prior to the request.

```

1016 \def\pcol@shiftspanning#1{%
1017   \ifpcol@swapcolumn\ifodd\c@page\else
1018     \setbox#1\ vbox{\@tempdima\textwidth \advance\@tempdima-\columnwidth
1019       \moveleft\@tempdima\box#1}
1020   \fi\fi}
1021

```

$\backslash\text{pcol@restartcolumn}$ The macro $\backslash\text{pcol@restartcolumn}$ is invoked from $\backslash\text{pcol@output@switch}$ or $\backslash\text{pcol@freshpage}$ to restart the current column-page $d = \backslash\text{pcol@nextcol}$ which becomes $c = \backslash\text{pcol@currcol}$ by the very first assignment in this macro. Then we restore the column-context in κ_c by $\backslash\text{pcol@getcurrcol}$ and let $p = \kappa_c(\beta^p)$ and $\backslash\text{@colroom} = \kappa_c(\beta^r)$ before returning $\kappa_c(\beta)$ to $\backslash\text{@freelist}$ by $\backslash\text{@cons}$ because it has become useless so far. We also restore the page context of p by $\backslash\text{pcol@getcurrpage}$.

```

1022 \def\pcol@restartcolumn{%
1023   \global\pcol@currcol\pcol@nextcol
1024   \pcol@getcurrcol
1025   \global\pcol@page\count\@currbox
1026   \global\@colroom\dimen\@currbox
1027   \pcol@Fb
1028   \@cons\@freelist\@currbox
1029   \pcol@Fe{restartcolumn(col)}%
1030   \pcol@getcurrpage

```

Then we perform footnote $\backslash\text{insertion}$ as follows.

- (1) If footnote typesetting is page-wise and $p = p_t$, we do the followings.
 - (a) Put the contents of $\kappa_c(\beta^b)$ by $\backslash\text{pcol@putbackmv1}$ to make the color context in .dvi consistent with the current .tex 's one, and to save pre-spanning-text stuff into $\backslash\text{pcol@prespan}$ if we are opening a spanning text.
 - (b) Put $\backslash\text{penalty} = 10000$ by $\backslash\text{nobreak}$ if $\backslash\text{if@nobreak} = \text{true}$ or $\backslash\text{interlinepenalty}$ by $\backslash\text{addpenalty}$ ¹⁷⁴ otherwise, as the page break penalty at the returning point. Note that adding the $\backslash\text{penalty}$ will be nullified by $\text{T}_{\text{E}}\text{X}$ if $\kappa_c(\beta^b)$ has nothing and thus, if the column-page is still empty when we leave from it, its emptiness without any items is assured. Also note that the penalty insertion here *looks* essential to keep $\text{T}_{\text{E}}\text{X}$'s page builder from confusing with page-wise footnotes which it has not seen in a column-page¹⁷⁵.
 - (c) If $\pi^f(p) \neq \perp$, let $\backslash\text{pcol@currfoot}$ and then $\backslash\text{footins}$ have the footnotes in it by an $\backslash\text{edef}$ and $\backslash\text{pcol@getcurrfoot}$, return the $\backslash\text{insert}$ for them to $\backslash\text{@freelist}$, invoke $\backslash\text{pcol@shrinkcolbyfn}$ to shrink $\backslash\text{@colht}$ temporarily by their total height and to remember the existence of them with $\backslash\text{@tempdimb} = -\backslash\text{skip}\backslash\text{footins}$, and then $\backslash\text{insert}$ the footnotes so that it contributes to the building process of the column-page to be restarted. Otherwise, i.e. if $\pi^f(p) = \perp$, $\backslash\text{@colht}$ is unchanged and $\backslash\text{@tempdimb} = 0$.
 - (d) Invoke $\backslash\text{pcol@deferredfootins}$ to $\backslash\text{insert}$ deferred footnotes in Φ until their total height reaches (possibly shrunk) $\backslash\text{@colht}$. This height capping is to keep $\text{T}_{\text{E}}\text{X}$'s page

¹⁷⁴As done in $\backslash\text{@specialoutput}$ but $\backslash\text{penalty}\backslash\text{interlinepenalty}$ should be sufficient.

¹⁷⁵At least a test with tall page-wise footnotes gave us a confusing result.

builder from holding too large number of footnotes unprocessed causing confused ordering on presenting them to `\output` routine.

- (2) If footnote typesetting is page-wise but $p < p_t$, we do the followings.
- (a) If $\pi^f(p) \neq \perp$, get it into `\footins` as done in (1c) but giving `\copy` to `\pcol@getcurrfoot` because $\pi^f(p)$ has been fixed and thus will be kept until it is shipped out, and then `\insert` it.
 - (b) Put $\kappa_c(\beta^b)$ and the penalty as done in (1a) and (1b).

The order of footnotes, main vertical list and then penalty is essential to ensure that the column-page in $p < p_t$ has room for footnotes whose residence in p has already been fixed.

- (3) If footnote typesetting is column-wise, we do the followings.
- (a) Put $\kappa_c(\beta^b)$ as done in (1a).
 - (b) If $\kappa_c(ft) \neq \perp$, get it by `\pcol@getcurrfoot` returning the `\insert` to `\@freelist`, and then `\insert` it.
 - (c) Put a penalty as done in (1b).

The order of main vertical list, footnotes and then penalty is appropriate for column-wise footnotes because they definitely have space in the column-page and `TEX` will break the page below the insertion, possibly just below thanks to the penalty, to keep the footnotes and references to them in a page.

```

1031 \ifpcol@scfnote
1032 \edef\pcol@currfoot{\pcol@footins}%
1033 \ifnum\pcol@page=\pcol@toppage
1034 \@tempdima\@colht \@tempdimb\z@
1035 \pcol@putbackmvl
1036 \if@nobreak \nobreak \else \addpenalty\interlinepenalty \fi
1037 \ifvoid\pcol@footins\else
1038 \pcol@Fb
1039 \pcol@getcurrfoot\box \@cons\@freelist\pcol@currfoot
1040 \pcol@Fe{restartcolumn(pagefn)}%
1041 \pcol@Log\pcol@restartcolumn{insert}\footins
1042 \pcol@shrinkcolbyfn\@colht\footins\@tempdimb
1043 \insert\footins{\unvbox\footins}%
1044 \fi
1045 \pcol@deferredfootins\pcol@restartcolumn
1046 \@colht\@tempdima
1047 \else
1048 \ifvoid\pcol@footins\else
1049 \pcol@getcurrfoot\copy
1050 \pcol@Log\pcol@restartcolumn{insdmy}\footins
1051 \insert\footins{\unvbox\footins}%
1052 \fi
1053 \pcol@putbackmvl
1054 \if@nobreak \nobreak \else \addpenalty\interlinepenalty \fi
1055 \fi
1056 \else
1057 \pcol@putbackmvl
1058 \ifvoid\pcol@currfoot\else

```



```

1059 \pcol@Fb
1060 \pcol@getcurrfoot\box \@cons\@freelist\pcol@currfoot
1061 \pcol@Fe{restartcolumn(colfn)}%
1062 \pcol@Log\pcol@restartcolumn{insert}\footins
1063 \insert\footins{\unvbox\footins}%
1064 \fi
1065 \if@nobreak \nobreak \else \addpenalty\interlinepenalty \fi
1066 \fi}
1067

```

`\pcol@getcurrcol` The macro `\pcol@getcurrcol` is invoked from the following macros to restore the typesetting parameters of the column $c = \pcol@currcol$ from κ_c , and to let `\columnwidth` have $w_c = \pcol@iigetcurrcol \pcol@columnwidth \cdot c$ ¹⁷⁶.

<code>\pcol@output@start</code>	<code>\pcol@output@switch</code>	<code>\pcol@restartcolumn</code>
<code>\pcol@flushcolumn</code>	<code>\pcol@measurecolumn</code>	<code>\pcol@synccolumn</code>
<code>\pcol@makeflushedpage</code>	<code>\pcol@imakeflushedpage</code>	<code>\pcol@iflushfloats</code>
<code>\pcol@freshpage</code>	<code>\pcol@output@end</code>	

Since we represent κ_c as;

$$\{\kappa_c(\beta)\}\{\kappa_c(\tau)\}\{\kappa_c(\delta)\}\{\kappa_c(\lambda_t)\}\{\kappa_c(\lambda_m)\}\{\kappa_c(\lambda_b)\}\{\kappa_c(\lambda_d)\}\{\kappa_c(\xi)\}\% \\ \{\kappa_c(\eta)\}\{\kappa_c(\nu_t)\}\{\kappa_c(\rho_t)\}\{\kappa_c(\nu_b)\}\{\kappa_c(\rho_b)\}\{\kappa_c(\nu_c)\}\{\kappa_c(\sigma)\}\{\kappa_c(\varepsilon)\}\}$$

in the body of `\pcol@col·c`, we restore first eight by `\pcol@iigetcurrcol` giving everything above as its arguments by the expansion of

$$\backslash\csname pcol@col\number\pcol@currcol\endcsname$$

and then of the resulting control sequence. Then this macro gives its ninth argument to `\pcol@iigetcurrcol` which restores the last eight. We also do

$$\backslash\global\columnwidth\pcol@columnwidth \cdot c$$

by a pair of `\expandafter` for the first two control sequences.

Note that the restore operations are `\global`, except for $\kappa_c(\beta)$ and $\kappa_d(\tau)$ because they are referred to only in `\output`, including `\if@nobreak` for which `\@nobreaktrue` and `\@nobreakfalse` are defined `\global` by L^AT_EX. Also note that `\dimen`-type parameters are saved in the form of integers and thus restoring them needs to specify the unit `sp`.

```

1068 \def\pcol@getcurrcol{%
1069 \expandafter\expandafter\expandafter\pcol@iigetcurrcol
1070 \csname pcol@col\number\pcol@currcol\endcsname
1071 \expandafter\global\expandafter\columnwidth
1072 \csname pcol@columnwidth\number\pcol@currcol\endcsname}
1073 \def\pcol@iigetcurrcol#1#2#3#4#5#6#7#8#9{%
1074 \def\@currbox{#1}\def\pcol@currfoot{#2}\global\pcol@prevdepth#3sp\relax
1075 \gdef\@toplist{#4}\gdef\@midlist{#5}\gdef\@botlist{#6}\gdef\@deferlist{#7}%
1076 \global\pcol@textfloatsep#8sp\pcol@iigetcurrcol#9}
1077 \def\pcol@iigetcurrcol#1#2#3#4#5#6#7#8{%
1078 \global\@textfloatsheight#1sp\relax
1079 \global\@topnum#2\relax \global\@toproom#3sp\relax
1080 \global\@botnum#4\relax \global\@botroom#5sp\relax

```

¹⁷⁶`\hsize` and `\linewidth` are let have w_c and $w_c - \mu$ respectively in `\pcol@invokeoutput`.

```

1081 \global\@colnum#6\relax
1082 \global\@afterindentfalse \@nbreaktrue
1083 \ifcase#7
1084   \@nbreakfalse \or
1085   \global\@afterindenttrue \else
1086   \relax
1087 \fi
1088 \global\everypar{#8}}

```

`\pcol@getcurrfoot` The macro `\pcol@getcurrfoot` $\langle com \rangle$ is invoked from `\pcol@startcolumn` ($\backslash copy$, $\pi^f(p)$), `\pcol@restartcolumn` ($\backslash copy/\backslash box$, $\pi^f(p)/\kappa_c(\tau)$), `\pcol@flushcolumn` ($\backslash box$, $\kappa_c(\tau)$) and `\pcol@imakeflushedpage` ($\backslash box$, $\kappa_c(\tau)$) to put everything in `\pcol@currfoot`, having the second element in the parens following macro names, into `\footins` using $\langle com \rangle$ shown as the first element in parens for the $\backslash box$ component. That is, if the source $\pi^f(p)$ or $\kappa_c(\tau^b)$ is void, we let $\backslash box.\backslash footins$ be so. Otherwise, we move $\backslash box$, $\backslash count$, $\backslash dimen$ and $\backslash skip$ of the source into those of $\backslash footins$ ¹⁷⁷.

```

1089 \def\pcol@getcurrfoot#1{%
1090   \ifvoid\pcol@currfoot \global\setbox\footins\box\voidb@x
1091   \else
1092     \global\setbox\footins#1\pcol@currfoot
1093     \global\count\footins\count\pcol@currfoot
1094     \global\dimen\footins\dimen\pcol@currfoot
1095     \global\skip\footins\skip\pcol@currfoot
1096   \fi}

```

`\pcol@setcurrcol` and `\pcol@setcurrcolnf` The macro `\pcol@setcurrcol` is invoked from `\pcol@output@switch`, `\pcol@measurecolumn` and `\pcol@synccolumn` to save column-context of $c = \backslash pcol@currcol$ in κ_c . It is also used in `\pcol@setcurrcolnf` invoked from `\pcol@output@start`, `\pcol@flushcolumn`, `\pcol@imakeflushedpage`, `\pcol@iflushfloats` and `\pcol@freshpage` for the saving when the column-page is known to have no footnotes.

The macro `\pcol@setcurrcol` at first calculates the combined code for `\if@nbreak` and `\if@afterindent`, and then saves parameters into κ_c by `\xdef` to have the sequence shown in the description of `\pcol@getcurrcol`. Note that $\backslash dimen$ -type parameters are saved by expansions with `\number` and thus as decimal integers.

The macro `\pcol@setcurrcolnf` defines $\kappa_c(\tau) = \backslash pcol@currfoot$ as $\backslash voidb@x$, and then invoke `\pcol@setcurrcol` for saving.

```

1097 \def\pcol@setcurrcol{\let\@elt\relax
1098   \@tempcnta\if@nbreak\if@afterindent\@ne\else\tw@\fi\else\z@\fi
1099   \expandafter\xdef\csname pcol@col\number\pcol@currcol\endcsname{%
1100     {\@currbox}{\pcol@currfoot}{\number\pcol@prevdepth}%
1101     {\@toplist}{\@midlist}{\@botlist}{\@deferlist}{\number\pcol@textfloatsep}%
1102     {\number\@textfloatsheight}%
1103     {\number\@topnum}{\number\@toproom}{\number\@botnum}{\number\@botroom}%
1104     {\number\@colnum}{\number\@tempcnta}{\the\everypar}}}}
1105 \def\pcol@setcurrcolnf{\def\pcol@currfoot{\voidb@x}\pcol@setcurrcol}
1106

```

`\pcol@putbackmvl` The macro `\pcol@putbackmvl`, solely used in `\pcol@restartcolumn`, has two functions; color stack restoration and pre-spanning-text stuff preservation. It examines the emptiness of the

¹⁷⁷Moving $\backslash count$, $\backslash dimen$ and $\backslash skip$ is redundant almost always because it is very unlikely that these footnote parameters are modified dynamically. Moreover, dynamic modification of them is hardly consistent with repetitive self- $\backslash insert$ of $\backslash footins$ in `\pcol@restartcolumn` and `\@reinserts` of L^AT_EX. However, we dare to move them in order to, for example, allow each column has its own footnote parameters.

column-page of the column c to be restarted in $\kappa_c(\beta) = \backslash@currbox$. If so, the color stack Γ^c is saved into $\Gamma_s = \backslash\text{pcol@colorstack@saved}$ by $\backslash\text{pcol@savecolorstack}$ as the opening color context of the column-page, and $\backslash\text{pcol@prespan}$ for pre-spanning-text stuff is made \perp .

Otherwise, Γ_s is let \perp because the opening color context has already been put when we left from the column-page. Then if $\backslash\text{ifpcol@sptextstart} = \text{true}$ to mean a spanning text is to start, we save $\kappa_c(\beta)$ into $\backslash\text{pcol@prespan}$ adding the coloring $\backslash\text{specials}$ to restore color context from Γ^c by $\backslash\text{pcol@restorecolorstack}$ ¹⁷⁸. We also shrink $\kappa_c(\beta^r) = \backslash@colroom$ by the height of the pre-spanning-text stuff so that spanning text will be captured by $\backslash\text{pcol@makecol}$ if it is broken into two (or more) pages, and put a invisible $\backslash\text{hrule}$ to the main vertical list letting $\backslash\text{topskip} = 0$ to suppress $\backslash\text{topskip}$ insertion prior to the spanning text but instead to make the text led by $\backslash\text{baselineskip}$ (or $\backslash\text{lineskip}$) according to the $\backslash\text{prevdepth}$ being the depth of the tallest column and the height of the first $\backslash\text{hbox}$ in the spanning text. Otherwise, i.e., if $\backslash\text{ifpcol@sptextstart} = \text{false}$, we simply put back $\kappa_c(\beta)$ into the main vertical list and then the coloring $\backslash\text{specials}$ by $\backslash\text{pcol@restorecolorstack}$.

Note that $\backslash\text{ifpcol@sptextstart}$ is temporarily made *false* by this macro if the $\backslash\text{pcol@output@switch}$ invoking $\backslash\text{pcol@restartcolumn}$ did not make synchronized column-switching, i.e., if $\backslash\text{ifpcol@flush} = \text{true}$ to mean the page is flushed before the synchronization, or $\backslash\text{ifpcol@sync} = \text{false}$ for column-scanning prior to the synchronization.

```

1107 \def\pcol@putbackmvl{%
1108   \ifpcol@flush \pcol@sptextstartfalse \fi
1109   \ifpcol@sync\else \pcol@sptextstartfalse \fi
1110   \pcol@ifempty\@currbox
1111   {\pcol@savecolorstack
1112    \ifpcol@sptextstart \global\setbox\pcol@prespan\box\voidb@x \fi}%
1113   {\global\setbox\pcol@colorstack@saved\box\voidb@x
1114    \ifpcol@sptextstart
1115     \global\setbox\pcol@prespan\vbox{%
1116       \unvbox\@currbox \pcol@restorecolorstack}%
1117     \global\advance\@colroom-\ht\pcol@prespan
1118     \global\topskip\z@ \hrule\@height\z@\@width\z@
1119     \else
1120     \unvbox\@currbox \pcol@restorecolorstack
1121     \fi}}
1122
```

11.4 Color Management

$\backslash\text{pcol@magicpenalty}$ The macro $\backslash\text{pcol@ifempty}\langle\text{box}\rangle\langle\text{then}\rangle\langle\text{else}\rangle$ is used in $\backslash\text{pcol@putbackmvl}$, $\backslash\text{pcol@clearcst@}$
 $\backslash\text{pcol@ifempty}$ unvbox and $\backslash\text{pcol@measurecolumn}$ to examine if $\langle\text{box}\rangle$ is empty, and to perform $\langle\text{then}\rangle$ if
 so or $\langle\text{else}\rangle$ otherwise. Since T_EX does not provide any convenient way for the examination
 unfortunately, we perform a series of tricky operations to put the followings into $\backslash@tempboxa$;
 a penalty of $\backslash\text{pcol@magicpenalty} = 12345$ whose existence in the $\langle\text{box}\rangle$ is (almost) impossible;
 contents of $\langle\text{box}\rangle$ put by $\backslash\text{unvcopy}$; and then a $\backslash\text{global}$ definition of $\backslash@gtempa$ to let it have
 the decimal representation of $\backslash\text{lastpenalty}$. Since $\backslash\text{lastpenalty}$ has $\langle\text{pen}\rangle$ if the last item is
 $\backslash\text{penalty}\langle\text{pen}\rangle$, or 0 otherwise, $\backslash@gtempa = \backslash\text{pcol@magicpenalty}$ iff $\langle\text{box}\rangle$ is empty.

```

1123 %% Special Output Routines: Color Management
1124
```

¹⁷⁸If $\backslash\text{pcol@prespan}$ is connected to (the first part of) the spanning text, the reestablishment of the color stack here correctly places coloring $\backslash\text{specials}$ in .dvi . On the other hand, if the spanning text is stowed away to the next page as a whole, the reestablishment here is essential for the correct paring of the pushes and pops, the latter of which are at the bottom of the column-page whose tail is $\backslash\text{pcol@prespan}$.

```

1125 \def\pcol@magicpenalty{12345}
1126 \def\pcol@ifempty#1#2#3{%
1127   \setbox\@tempboxa\vbox{\penalty\pcol@magicpenalty
1128     \unvcopy#1\xdef\@gtempa{\number\lastpenalty}}}%
1129 \ifnum\@gtempa=\pcol@magicpenalty\relax \def\reserved@a{#2}%
1130 \else \def\reserved@a{#3}%
1131 \fi
1132 \reserved@a}
1133

```

`\pcol@clearcst@unvbox` The macro `\pcol@clearcst@unvbox` $\langle box \rangle$, invoked from `\pcol@opcol` and `\pcol@output@`
`\pcol@clearcolorstack` switch, puts the following above and below $\langle box \rangle$ containing the main vertical list of a column-page from which we are now leaving, if `\pcol@ifempty` judges that $\langle box \rangle$ is not empty. Above the $\langle box \rangle$, we put coloring `\specials` to establish the color stack of `.dvi` saved in $\Gamma_s = \pcol@colorstack@sav$ ed by `\pcol@restorecst` as the opening color context of the column-page. The stack Γ_s , however, can be \perp if $\langle box \rangle$ already has the `\specials`, i.e., when we visit the column-page it had already had some items. Below $\langle box \rangle$, on the other hand, we put uncoloring `\specials` by `\pcol@clearcolorstack` to rewind Γ_r^c to clear the color context of the column-page in `.dvi` temporarily so that afterward it is made consistent with that in `.tex`.

The macro `\pcol@clearcolorstack`, solely invoked from `\pcol@clearcst@unvbox` shown above¹⁷⁹, scans $\Gamma_r^c = (\gamma_0^c, \Gamma_r)$ by `\pcol@scancst` giving $\Gamma_r = \pcol@colorins$ to it as its argument. Since we gives Γ_r to the macro, this scan includes removals of all γ_i^- and $\gamma_{i,m}^-$, all γ_i having matching γ_i^- , all $\gamma_{i,m}$ having matching $\gamma_{i,m}^-$, and all elements to update γ_0^c , from Γ_r to let `\pcol@colorins` have Γ . Prior to this invocation, we `\define \reserved@a` $\langle \gamma_i \rangle$ and `\reserved@b` $\langle \gamma_0^c \rangle$ to let them have `\reset@color` so that uncoloring `\special` will be put into the main vertical list for each $\gamma_i \in \Gamma$ and γ_0^c before update if any, regardless of coloring `\special` they have. That is, we invoke `\reset@color` as many times as the appearance of $\gamma_i \in \Gamma$ and once if $\gamma_0^c \neq \perp$ before the invocation, ignoring the color information in each element and the order of elements, expecting `\reset@color` just pops printer's color stack to rewind it as we intend.

Note that in some printer `.definition` could `\define \reset@color` to let printer's text color be `\current@color` to make the stack rewinding resulting in the sequence of coloring operations with `\current@color` at the invocation of `\output`. This meaningless operations might cause a problem when a colored column-page of c_1 is physically followed by another column-page of the succeeding column c_2 without any coloring, because the column-page of c_2 will be colored with `\current@color` at the page break in c_1 . If this problem is serious, we could initialize γ_0^c with `\current@color` at `\begin{paracol}` for all c such that $\hat{\gamma}_0^c$ is undefined, in order to make sure that any column-page has at least one coloring `\special` with γ_0^c at its beginning so that, for example, coloring operations at the tail of the column-page of c_1 is overridden by that of the default color of c_2 placed at the head of its column-page.

```

1134 \def\pcol@clearcst@unvbox#1{%
1135   \pcol@ifempty#1\relax
1136   {\pcol@restorecst\pcol@colorstack@sav \unvbox#1\pcol@clearcolorstack}}
1137 \def\pcol@clearcolorstack{%
1138   \def\reserved@a##1{\reset@color}\def\reserved@b##1{\reset@color}%
1139   \pcol@scancst\pcol@colorins}
1140

```

¹⁷⁹But we have this macro to avoid the complication in `\defining \reserved@a` and `\reserved@b` with an argument if we did it in the argument of `\pcol@ifempty` in `\pcol@clearcst@unvbox`.

`\pcol@restorecolorstack` The macro `\pcol@restorecolorstack`, used in `\pcol@putbackmvl` and `\pcol@output@end`,
`\pcol@restorecst` makes color context in `.dvi` consistent with that in `.tex` by giving $\Gamma = \pcol@colorins$ to
`\pcol@restorecst` to let it scan $\Gamma^c = (\gamma_0^c, \Gamma)$. The callee macro `\pcol@restorecst<box>`,
also used in `\pcol@clearcst@unvbox` with $\langle box \rangle = \Gamma_s = \pcol@colorstack@saved$, invokes
`\pcol@scancst` after `\defining \reserved@a< γ_i >` to apply `\unvbox` to γ_i in Γ or Γ_s and
`\reserved@b< γ_0^c >` to apply `\unvcopy` to γ_0^c so that coloring `\specials` they have will be put
into the main vertical list.

```

1141 \def\pcol@restorecolorstack{\pcol@restorecst\pcol@colorins}
1142 \def\pcol@restorecst{%
1143   \def\reserved@a##1{\unvbox##1}\def\reserved@b##1{\unvcopy##1}%
1144   \pcol@scancst}
1145

```

`\pcol@scancst` The macro `\pcol@scancst<box>` is invoked from `\pcol@clearcolorstack` and `\pcol@`
`\pcol@iscancst` `restorecst`. In the former invocation, we have $\langle box \rangle = \Gamma_r = \pcol@colorins$ to rewind
 $\Gamma_r = (\gamma_0^c, \Gamma_r)$ with `\reserved@a< γ_i >` and `\reserved@b< γ_0^c >` having `\reset@color`. In the lat-
ter one, we have $\langle box \rangle \in \{\Gamma = \pcol@colorins, \Gamma_s = \pcol@colorstack@saved\}$ to reestablish
 $\Gamma^c = (\gamma_0^c, \Gamma)$ or Γ_s with `\reserved@a< γ_i >` to apply `\unvbox` to γ_i and `\reserved@b< γ_0^c >` to apply
`\unvcopy` to γ_0^c . Therefore, if $\langle box \rangle = \pcol@colorins$, we at first put (un)coloring `\special`
for γ_0^c , unless it is \perp , to the main vertical list applying `\reserved@b` to it. This means the
`\special` for γ_0^c is put first prior to those for elements in Γ_r or Γ consistently in reestablishing
but not in rewinding. However as we discussed in the description of `\pcol@clearcolorstack`,
the order of rewinding does not affect the result for almost all printers because only the number
of pop operations is significant for them¹⁸⁰. Then if $\langle box \rangle \neq \perp$ we invoke `\pcol@iscancst`
to examine the contents of $\langle box \rangle$ from its bottom to top. Prior to the invocation, we do the fol-
lowing; let `\@tempboxa` have an empty `\vbox` as its initial value of reformed $\langle box \rangle$; let `\pcol@`
`tempboxa` have an empty `\vbox` as its initial value of the sequence of `\specials` to be put into
the main vertical list; let $n_{\text{pop}} = \@tempcntb = 0$; let $M = \reserved@b = ()$ as its initial
value of the list of identifiers of math-mode pops; `\if@tempswa = true` to mean the first `\vbox`
to update γ_0^c found in the scan (i.e., the bottommost one) is effective.

In the macro `\pcol@iscancst`, we repeatedly examine the last `\vbox` in $\langle box \rangle$ taken by
`\lastbox` into $\gamma = \pcol@tempboxb$ until γ becomes \perp , and perform one of the following for
 γ .

- (1) If $height(\gamma) = 0$ and $width(\gamma) = 0$ to mean $\gamma = \gamma_i^-$, increment n_{pop} .
- (2) If $height(\gamma) = 0$ and $width(\gamma) = m > 0$ to mean $\gamma = \gamma_{i,m}^-$, let $M = (m, M)$.
- (3) If $height(\gamma) \neq 0$, $depth(\gamma) = 0$ and $width(\gamma) = 0$ to mean $\gamma = \gamma_i$, decrement n_{pop} if
 $n_{\text{pop}} > 0$, or otherwise add γ to the head of `\@tempboxa` and apply `\reserved@a< γ >` to
add its result to the head of `\pcol@tempboxa`.
- (4) If $height(\gamma) \neq 0$, $depth(\gamma) = 0$ and $width(\gamma) = m > 0$ to mean $\gamma = \gamma_{i,m}$, do nothing if
 $m \in M$, or otherwise add γ to the head of `\@tempboxa` and apply `\reserved@a< γ >` to
add its result to the head of `\pcol@tempboxa`.
- (5) If $height(\gamma) \neq 0$, $depth(\gamma) \neq 0$ to mean γ has a `\special` with which γ_0^c is updated.
If `\if@tempswa = true` to mean γ is the first (bottommost) occurrence, γ_0^c is updated
acquiring an `\insert` from `\@freelist` if it was \perp . In this case of \perp , we have to put
an uncoloring `\special` by `\reset@color`, because `\pcol@scancst` did not do it but

¹⁸⁰And even for the minority because multiple updates of printer's color with one particular color are inde-
pendent of the order of them.

`\columncolor` or `\normalcolumncolor` pushed the corresponding color `\special`. Then we let `\if@tempswa = false`.

Otherwise, i.e., if `\if@tempswa = false` for second or succeeding occurrences, we do nothing because updates by them are overridden by the first one.

Note that the cases other than (3) and (4) happen only in rewinding, and thus in reestablishing we only have (3) and (4) with $n_{\text{pop}} = 0$ and $M = ()$ always so that every γ is kept into new $\langle \text{box} \rangle$ and a coloring `\special` for it will be put into the main vertical list.

Then we go back to `\pcol@scancst` to let $\langle \text{box} \rangle = \text{\@tempboxa}$, meaningless in reestablishing but not harmful, and put the contents of `\pcol@tempboxa` to the main vertical list.

```

1146 \def\pcol@scancst#1{%
1147   \@tempcnta#1\relax
1148   \ifnum\@tempcnta=\pcol@colorins
1149     \ifvoid\pcol@ccuse{@box}\else
1150       \reserved@b{\pcol@ccuse{@box}}\fi
1151   \fi
1152   \ifvoid\@tempcnta\else
1153     \setbox\@tempboxa\vbox{}\setbox\pcol@tempboxa\vbox{}\@tempcntb\z@
1154     \def\reserved@b{\let\@elt\relax \@tempswatrue \pcol@iscancst
1155       \global\setbox\@tempcnta\box\@tempboxa \unvbox\pcol@tempboxa
1156     \fi}
1157 \def\pcol@iscancst{%
1158   \setbox\@tempcnta\vbox{%
1159     \unvbox\@tempcnta \global\setbox\pcol@tempboxb\lastbox}%
1160   \ifvoid\pcol@tempboxb \let\reserved@c\relax
1161   \else
1162     \let\reserved@c\pcol@iscancst
1163     \ifdim\ht\pcol@tempboxb=\z@
1164       \ifdim\wd\pcol@tempboxb=\z@ \advance\@tempcntb\@ne
1165       \else \edef\reserved@b{\@elt{\number\wd\pcol@tempboxb}\reserved@b}%
1166       \fi
1167     \else\ifdim\dp\pcol@tempboxb=\z@
1168       \ifdim\wd\pcol@tempboxb=\z@
1169         \ifnum\@tempcntb>\z@ \advance\@tempcntb\m@ne
1170         \else
1171           \setbox\@tempboxa\vbox{\copy\pcol@tempboxb \unvbox\@tempboxa}%
1172           \setbox\pcol@tempboxa\vbox{%
1173             \reserved@a\pcol@tempboxb \unvbox\pcol@tempboxa}%
1174           \fi
1175         \else
1176           \count@\wd\pcol@tempboxb \chardef\reserved@d\z@
1177           \def\@elt##1{\ifnum##1=\count@ \chardef\reserved@d\@ne \fi}%
1178           \reserved@b \let\@elt\relax
1179           \ifnum\reserved@d=\z@
1180             \setbox\@tempboxa\vbox{\copy\pcol@tempboxb \unvbox\@tempboxa}%
1181             \setbox\pcol@tempboxa\vbox{%
1182               \reserved@a\pcol@tempboxb \unvbox\pcol@tempboxa}%
1183             \fi
1184           \fi
1185         \else\if@tempswa
1186           \ifvoid\pcol@ccuse{@box}%
1187           \@next\@currbox\@freelist{\global\setbox\@currbox\vbox{}}\pcol@ovf
1188           \pcol@ccxdef{\@currbox}\reset@color

```

```

1189     \fi
1190     \global\setbox\pcol@ccuse{@box}\vbox{\unvbox\pcol@tempboxb}%
1191     \@tempswafalse
1192     \fi\fi\fi
1193 \fi
1194 \reserved@c}
1195

```

`\pcol@savestack` The macro `\pcol@savestack` is used in `\pcol@startcolumn`, `\pcol@output@start` and `\pcol@putbackmv1` to save the opening color context in Γ^c of a current column-page c known to be or found empty into $\Gamma_s = \text{\pcol@colorstack@saved}$. If both of $\gamma_0^c = \text{\pcol@columncolor@box} \cdot c$ and $\Gamma = \text{\pcol@colorins}$ are \perp , Γ_s is let be \perp . Otherwise, Γ_s is let be a `\vbox` having a `\vbox` for γ_0^c at the top if it is not \perp and then the contents of Γ if it is not \perp .

```

1196 \def\pcol@savestack{%
1197   \ifvoid\pcol@colorins \@tempswafalse \else \@tempwatruer \fi
1198   \ifvoid\pcol@ccuse{@box}%
1199     \setbox\@tempboxa\box\voidb@x
1200   \else
1201     \setbox\@tempboxa\vbox{\unvcopy\pcol@ccuse{@box}}%
1202     \ht\@tempboxa1sp \dp\@tempboxa\z@ \wd\@tempboxa\z@
1203     \@tempwatruer
1204   \fi
1205   \if@tempswa
1206     \global\setbox\pcol@colorstack@saved\vbox{%
1207       \ifvoid\@tempboxa\else \box\@tempboxa \fi
1208       \ifvoid\pcol@colorins\else \unvcopy\pcol@colorins \fi}
1209   \else \global\setbox\pcol@colorstack@saved\box\voidb@x
1210   \fi}
1211

```

`\pcol@ccuse` The macro `\pcol@ccuse<px>` is to expand a macro `\pcol@columncolor·<px>·c` for the current column c . It is used in `\pcol@output@start` and `\pcol@scancst@shadow` with `<px> = ‘` to have $\hat{\gamma}_0^c$, and in `\pcol@scancst`, `\pcol@iscancst`, `\pcol@savestack`, `\pcol@output@end` and `\pcol@icolumncolor` with `<px> = @box` to have γ_0^c .

The macro `\pcol@ifccdefined<then><else>` is used in `\pcol@output@start` and `\pcol@scancst@shadow` to examine whether $\hat{\gamma}_0^c = \text{\pcol@columncolor} \cdot c$ is defined and to do `<then>` if so or `<else>` otherwise.

The macro `\pcol@ccxdef<body>` `\xdefines` a macro $\gamma_0^c = \text{\pcol@columncolor@box} \cdot c$ as `<body>` for the current column c . It is used in `\pcol@output@start`, `\pcol@iscancst` and `\pcol@icolumncolor` with `<body> = \@currbox` when an `\insert` for γ_0^c is acquired, and in `\pcol@output@end` with `<body> = \voidb@x` after releasing γ_0^c to `\@freelist`.

```

1212 \def\pcol@ccuse#1{\@nameuse{pcol@columncolor#1\number\pcol@currcol}}
1213 \def\pcol@ifccdefined#1#2{%
1214   \expandafter\ifx\csname pcol@columncolor\number\pcol@currcol\endcsname\relax
1215   #2\else#1\fi}
1216 \def\pcol@ccxdef#1{%
1217   \expandafter\xdef
1218   \csname pcol@columncolor@box\number\pcol@currcol\endcsname{#1}}
1219

```

11.5 Footnote Handling

`\pcol@savefootins` The macro `\pcol@savefootins⟨cs⟩`, invoked from `\pcol@makecol` for page-wise footnotes and from `\pcol@output@switch` for both column-wise and page-wise footnotes, saves `\footins` to an `\insert` register obtained by `\@next` from `\@freelist`, and `\defines⟨cs⟩` being `\pcol@currfoot` or `\pcol@footins` so that it has the register as its body and the register is then saved into $\pi^f(p)$ or $\kappa_c(\tau)$. We save not only `\box` component of `\footins` but also `\count`, `\dimen` and `\skip`¹⁸¹.

```
1220 %% Special Output Routines: Footnote Handling
1221
1222 \def\pcol@savefootins#1{%
1223   \@next#1\@freelist{%
1224     \global\setbox#1\box\footins
1225     \global\count#1\count\footins
1226     \global\dimen#1\dimen\footins
1227     \global\skip#1\skip\footins}{\def#1{\voidb{x}\pcol@ovf}}
1228
```

`\pcol@shrinkcolbyfn` The macro `\pcol@shrinkcolbyfn⟨height⟩⟨ins⟩⟨skip⟩`, invoked from `\pcol@makecol`, `\pcol@ioutputelt`, `\pcol@startcolumn`, `\pcol@restartcolumn`, `\pcol@flushcolumn` and `\pcol@makeflushedpage`, shrinks $\langle height \rangle \in \{\@colht, \@tempdima\}$ temporarily so that a column or the set of all columns resides in a page with page-wise footnotes in the `\insert⟨ins⟩`. The shrinkage is calculated by adding the sum of the height plus depth of all footnotes, i.e., that of $\langle ins \rangle$, and the natural component of `\skip⟨ins⟩`, i.e., the vertical space inserted between the columns and the footnotes. Note that the stretch and shrink components of `\skip⟨ins⟩` cannot be incorporated into the calculation but their contribution to each column-page is taken care of by the following macro `\pcol@unvbox@cc1v` if required. Also note that the inverse of `\skip⟨ins⟩` is kept in $\langle skip \rangle$ if it is not `\relax` so that `\pcol@deferredfootins` knows $\langle ins \rangle \neq \perp$ when this macro is invoked from `\pcol@startcolumn` and `\pcol@restartcolumn` with $\langle skip \rangle = \@tempdimb$.

```
1229 \def\pcol@shrinkcolbyfn#1#2#3{%
1230   \ifx#3\relax\else #3-\skip#2\relax \fi
1231   \advance#1-\ht#2\advance#1-\dp#2\advance#1-\skip#2\relax}
```

`\pcol@unvbox@cc1v` The macro `\pcol@unvbox@cc1v⟨ins⟩`, invoked from `\pcol@makecol` and `\pcol@flushcolumn` when they work on a column-page with page-wise footnotes, adds the stretch and shrink components of `\skip⟨ins⟩` at the end of `\box255`, where $\langle ins \rangle$ is non-void $\pi^f(p)$ having page-wise footnotes. Before the addition, the macro goes back to the baseline of `\box255`¹⁸² to nullify the baseline progress mechanism so as to make it sure the exact amount of the vertical skip is added. Then it adds the stretch and shrink by at first adding the skip itself and then the negative amount of its natural component.

```
1232 \def\pcol@unvbox@cc1v#1{%
1233   \@tempdima\dp\@cc1v \unvbox\@cc1v
1234   \vskip \ifdim\@tempdima>\@maxdepth -\@maxdepth \else -\@tempdima \fi
1235   \vskip\skip#1\@tempdima\skip#1\vskip-\@tempdima}
1236
```

`\pcol@deferredfootins` The macro `\pcol@deferredfootins⟨macro⟩`, invoked from `\pcol@startcolumn` and `\pcol@restartcolumn`, tries to `\insert` some of leading deferred footnotes in Φ through `\footins`¹⁸³.

¹⁸¹Knowing these three components are virtually constants.

¹⁸²The comparison of the depth of `\box255` and `\@maxdepth` and taking the latter if it is smaller is really just-in-case.

¹⁸³The argument $\langle macro \rangle$ has the invoker itself shown in debug messages.

In order to avoid that `\footins` has footnotes across three or more pages to make confusion in the order of footnotes kept inside of $\text{T}_{\text{E}}\text{X}$, we cap the total height of footnotes by $h = \text{\@colht}$ if it has already shrunk by non-deferred footnotes in the page we are working on indicated by $\text{\@tempdimb} < 0$, or $h = \text{\@colht} - \text{\skip\footins}$ if the page does not have non-deferred footnotes indicated by $\text{\@tempdimb} = 0$. That is, we extract leading elements of Φ by `\vsplit` until their total height reaches h and, if some elements are obtained in `\@tempboxa`, `\insert` them through `\footins`. As for the remaining elements if any, we add a leading `\penalty\interlinepenalty` which should have been removed by `\vsplit`.

Note that we temporarily let `\splitmaxdepth = \@maxdepth`, `\splittopskip = 0` and `\vbadness = ∞` at the `\vsplit` so that the depth of the split first half is `\@maxdepth` at deepest, the second half does not have any skip at its top, and $\text{T}_{\text{E}}\text{X}$ will not complain of (almost) inevitable underfull. Also note that the successful extraction of the leading elements is examined by checking `\ht\@tempboxa` and thus we need to `\unvbox` it in itself because `\vsplit` makes the height h regardless of its contents.

```

1237 \def\pcol@deferredfootins#1{%
1238   \ifdim\@tempdimb=\z@ \@tempdimb\@colht \advance\@tempdimb-\skip\footins
1239   \else \@tempdimb\@colht
1240   \fi
1241   \ifvoid\pcol@topfnotes\else \ifdim\@tempdimb>\z@
1242     \begingroup
1243       \splitmaxdepth\@maxdepth \splittopskip\z@ \vbadness\@M
1244       \setbox\@tempboxa\vsplit\pcol@topfnotes to\@tempdimb
1245       \ifvoid\pcol@topfnotes\else
1246         \global\setbox\pcol@topfnotes\vbox{\penalty\interlinepenalty
1247           \unvbox\pcol@topfnotes}%
1248       \fi
1249       \setbox\@tempboxa\vbox{\unvbox\@tempboxa}%
1250       \ifdim\ht\@tempboxa>\z@
1251         \pcol@Log#1{add}\@tempboxa
1252         \insert\footins{\unvbox\@tempboxa}%
1253       \fi
1254     \endgroup
1255   \fi\fi}
1256

```

`\pcol@combinefootins` The macro `\pcol@combinefootins`(b)(f), invoked solely from `\pcol@makenormalcol`¹⁸⁴, constructs the pre-environment stuff in `\@outputbox`, combining the stuff in the box b and pre-environment footnotes in the `\insert` f . It is almost equivalent to the `\else` part of `\@makecol` in which the main vertical list and column-wise footnotes are combined. The operation to put footnotes is almost equivalent to the second half of the `\else` part except that `\insert` is not always `\footins` but f and is done by `\pcol@putfootins` in which a null vertical skip is added after f but this skip is removed by `\unskip` after the invocation. The other difference is that the vertical space of `\belowfootnoteskip` is added if it is not 0¹⁸⁵ to have some space between non-merged pre-environment footnotes and the first lines in a `paracol` environment.

The null vertical skip is put for page-wise footnotes, for which the macro `\pcol@putfootins` is invoked from `\pcol@ioutputelt` and `\pcol@makeflushedpage`. Since we shrink the height of column-pages by the height-plus-depth of page-wise footnotes, the natural height of the box in which column-pages and page-wise footnotes are combined would be less than `\textheight`

¹⁸⁴And thus having the arguments $\langle b \rangle$ and $\langle f \rangle$ is unnecessary, but we keep this implementation to avoid unnecessary recoding from a development version.

¹⁸⁵We avoid null space insertion to minimize the difference from older versions in traced output.

due to the depth of the last footnote line if we simply made the footnotes the last items of the box. Though this shortage at most `\maxdepth` is expected to be covered by the stretch factor of `\skip\footins` without too large badness causing an underfull message¹⁸⁶, someday we could face an underfull with some unusual settings of `\maxdimen`, `\skip\footins` and/or `\vbadness`. Therefore, we put a null vertical skip so that the real bottom of the footnotes, instead of the last baseline, is placed at the baseline of the box, to make the natural height of the box is `\textheight` exactly. Note that this shifting page-wise footnotes up will not make last baselines of footnotes among pages unaligned, because the last line have a strut.

```

1257 \def\pcol@combinefootins#1#2{%
1258   \setbox\@outputbox\vbox{%
1259     \boxmaxdepth\@maxdepth
1260     \unvbox#1\relax
1261     \pcol@putfootins#2\unskip
1262     \ifdim\belowfootnoteskip=\z@\else \vskip\belowfootnoteskip \fi}}
1263 \def\pcol@putfootins#1{%
1264   \vskip\skip#1\relax
1265   \color@begingroup
1266     \normalcolor
1267     \footnoterule
1268     \unvbox#1\relax
1269   \color@endgroup \vskip\z@}
1270

```

11.6 Marginal Notes

`\@addmarginpar` The macro `\pcol@addmarginpar` is our own version of `\@addmarginpar`, which `\pcol@addmarginpar` `\zparacol` makes `\let`-equal to `\pcol@addmarginpar` keeping its original definition in `\pcol@addmarginpar`. Therefore, in a `paracol` environment, the `\output` request made by L^AT_EX's `\marginpar` in the column c and page p is processed by `\pcol@addmarginpar` through our own `\output` routine being `\pcol@output`, `\pcol@specialoutput` and L^AT_EX's `\@specialoutput`. What we do in this macro are as follows; determine the margin which a marginal note goes to; temporarily modify the parameter $m_w = \text{\marginparwidth}$ or $m_s = \text{\marginparsep}$ according to the margin and the column; determine the position to place the marginal note consulting $\pi^m(p) = \text{\pcol@mparbottom}$ obtained by `\pcol@getcurrpage`: and update $\pi^m(p)$ according to the position.

First, there are the following parameters to determine the margin, and thus the value of `\if@firstcolumn` referred to in L^AT_EX's `\@addmarginpar` and meaning left if *true* or right if *false*.

- (1) The macros `\pcol@mpthreshold@l` and `\pcol@mpthreshold@r` defined by `\marginparthreshold` give us the threshold of the column number to let columns less than it go to the left margin while those equal to or greater than it to the right, according to the parallel-page the column belongs to. Therefore, we let `\if@firstcolumn = true` iff $c < \text{\pcol@mpthreshold@l} \wedge c < C_L$ or $c < \text{\pcol@mpthreshold@r} \wedge c \geq C_L$, as the fundamental setting.
- (2) If `\ifpcol@swapmarginpar = true` because the specifier ‘m’ is given to `\twosided` explicitly or implicitly, the decision in (1) is reversed if $\text{page}(p) \bmod 2 = 0$, and then this decision is reversed again if $c \geq C_L$ and `\ifpcol@paired = false` to mean c is in a

¹⁸⁶With default settings of `\maxdimen = 5pt` and the stretch factor `4pt` of `\skip\footins`, the badness $100 \times (5/4)^3 \approx 195$ is significantly less than the default `\vbadness = 1000`.

non-paired right parallel-page. The second reversal is done because $page(p)$ is *common* to both left and right parallel-pages and is for the left page in usual cases without page number jumps. Therefore, $(page(p) \bmod 2)$ is for left parallel-pages and thus for right counterparts the decision should be made by $((page(p) + 1) \bmod 2)$ and thus the result should be reversed.

- (3) If `\ifreversemargin` is made *true* by L^AT_EX's `\reversemarginpar`, the decision made by (1) and then possibly reversed by (2) is finally reversed.

Second, we calculate

$$D = \text{\@tempdima} = \sum_{d=C^0}^{swap(c)-1} (w_{swap(d)} + g_{swap'(d)})$$

where $x' = swap(x)$ is given by `\pcol@swapcolumn` $\langle x \rangle \langle x' \rangle \langle C^0 \rangle \langle C^1 \rangle$ to let $x' = C^1 - (x - C^0) - 1$ if `\ifpcol@swapcolumn = true` and $page(p) \bmod 2 = 0$ or $x' = x$ otherwise, $swap'(x) = \text{\pcol@colsepid} \in \{swap(x) - 1, x\}$ according to swapped or not, $(C^0, C^1) \in \{(0, C_L), (C_L, C)\}$ according to $c < C_L$ or not, and w_x and g_x are the width of x -th column and gap given by `\pcol@columnwidth` $\cdot x$ and `\pcol@columnsep` $\cdot x$ respectively. That is, D is the distance from the left edge of the column c to that of leftmost column in the (parallel-) page in which c resides. Then if `\if@firstcolumn = true` to let the marginal note go to the left margin, we add D to $m_w = \text{\marginparwidth}$ so that L^AT_EX's `\@addmarginpar` being `\pcol@@addmarginpar` aligns the left edge of the marginal note at the point apart from the column's left edge by $(D + m_w) + m_s$ rather than $m_w + m_s$, or in other words $m_w + m_s$ apart from the left edge of the leftmost column. On the other hand if `\if@firstcolumn = false`, we add $W_T - (D + w_c)$ where W_T is `\textwidth`, being the distance from the right edge of the column c to that of the rightmost column, to $m_s = \text{\marginparsep}$ so that `\pcol@@addmarginpar` aligns the left edge of the marginal note at the point apart from the column's right edge by $W_T - (D + w_c) + m_s$ rather than m_s , or in other words m_s apart from the right edge of the rightmost column.

Third, we let `\pcol@marbox` be the first element of `\@currlist` obtained by `\@xnext` for the right marginal note if `\if@firstcolumn = false`, or `\@currbox` for the left marginal note otherwise. Then we let $t = \text{\@tempdima}$ be the distance from the top edge of the column to that of the marginal note, namely `\@pageht` minus the height of `\pcol@marbox` plus `\dimen\@currbox` being downward shift amount optionally given by `\marginnote`. We also let $h = \text{\@tempdimb}$ be the height-plus-depth of the box `\pcol@marbox` plus `\marginparpush`, or in other words the vertical space the marginal note requires. Then we give t and h to `\pcol@getmparbottom` to let `\@mparbottom` have the bottom of the existing marginal text below which we put the marginal text in `\pcol@marbox`, and to let `\pcol@mpblist` have the new list to be replaced with $M_{\{L,R\}}^{\{l,r\}}$ in $\pi^m(p)$.

Forth, we update $\pi^m(p)$ with the new list in `\pcol@mpblist` by a process similar to `\pcol@setpageno` but with `\pcol@setmpbelt` to scan the list of pages Π^+ .

Fifth, we shift down `\pcol@marbox` by `\dimen\@currbox` and, if the shift amount is greater than the height of the box and thus the height of shifting result is zero, decrement `\@mparbottom` by the amount to deceive L^AT_EX's `\@addmarginpar` into believing `\@mparbottom` is above the real one by the amount. In other words, by the decrement we let `\@addmarginpar` see the top edge of the shifted marginal note in the box, rather than that of the box itself, for the placement with `\@mparbottom`.

Finally, we invoke L^AT_EX's original `\@addmarginpar` being `\pcol@@addmarginpar` to put the marginal note according to `\if@firstcolumn`, temporarily modified `\marginparsep` and `\marginparwidth`, and `\@mparbottom`. Note that since `\pcol@zparacol` lets `\if@twocolumn`

= *true*, `\pcol@addmarginpar` determine the margin only by `\if@firstcolumn`. Also note that it can be `\@mparbottom > t` (before decrement with the positive shift amount) to mean the marginal note should be shifted down from its natural position, and if so `\pcol@addmarginpar` will give us a warning as the correct consequence of the placement.

```

1271 %% Special Output Routines: Marginal Notes
1272
1273 \def\pcol@addmarginpar{%
1274   \pcol@getcurrpage \@firstcolumntrue
1275   \ifnum\pcol@currcol<\pcol@ncolleft
1276     \let\reserved@a\z@ \let\reserved@b\pcol@ncolleft
1277     \ifnum\pcol@mpthreshold@l>\pcol@currcol\else \@firstcolumnfalse \fi
1278   \else
1279     \let\reserved@a\pcol@ncolleft \let\reserved@b\pcol@ncol
1280     \ifnum\pcol@mpthreshold@r>\pcol@currcol\else \@firstcolumnfalse \fi
1281   \fi
1282   \ifpcol@swapmarginpar
1283     \ifodd\c@page\else
1284       \if@firstcolumn \@firstcolumnfalse \else \@firstcolumntrue \fi
1285     \fi
1286     \ifpcol@paired\else \ifnum\pcol@currcol<\pcol@ncolleft\else
1287       \if@firstcolumn \@firstcolumnfalse \else \@firstcolumntrue \fi
1288     \fi\fi
1289   \fi
1290   \if@reversemargin
1291     \if@firstcolumn \@firstcolumnfalse \else \@firstcolumntrue \fi
1292   \fi
1293   \pcol@swapcolumn\pcol@currcol\count@\reserved@a\reserved@b
1294   \@tempdima\z@
1295   \@tempcnta\reserved@a \@whilenum\@tempcnta<\count@\do{%
1296     \pcol@swapcolumn\@tempcnta\@tempcntb\reserved@a\reserved@b
1297     \advance\@tempdima\csname pcol@columnwidth\number\@tempcntb\endcsname\relax
1298     \advance\@tempdima\csname pcol@columnsep\pcol@colsepid\endcsname\relax
1299     \advance\@tempcnta\@ne}%
1300   \if@firstcolumn \advance\marginparwidth\@tempdima
1301   \else
1302     \advance\marginparsep\textwidth \advance\marginparsep-\@tempdima
1303     \advance\marginparsep-\columnwidth
1304   \fi
1305   \expandafter\@xnext\@currlist\@@\pcol@marbox\@gtempa
1306   \if@firstcolumn\let\pcol@marbox\@currbox \fi
1307   \@tempdima\@pageht \advance\@tempdima-\ht\pcol@marbox
1308   \advance\@tempdima\dimen\@currbox
1309   \@tempdimb\ht\pcol@marbox \advance\@tempdimb\dp\pcol@marbox
1310   \advance\@tempdimb\marginparpush
1311   \pcol@getmparbottom\@tempdima\@tempdimb
1312   \begingroup
1313     \@tempcnta\pcol@page \advance\@tempcnta-\pcol@basepage
1314     \edef\reserved@a{\pcol@pages\pcol@currpage}%
1315     \global\let\pcol@pages\@empty \global\let\pcol@currpage\@empty
1316     \let\@elt\pcol@setmpbelt \reserved@a
1317   \endgroup
1318   \ifdim\dimen\@currbox=\z@\else
1319     \ifdim\dimen\@currbox>\ht\pcol@marbox
1320       \advance\@mparbottom-\dimen\pcol@marbox

```

```

1321   \fi
1322   \setbox\pcol@marbox\hbox{\lower\dimen\@currbox\box\pcol@marbox}%
1323   \fi
1324   \pcol@addmarginpar}
1325

```

`\pcol@getmparbottom` The macro `\pcol@getmparbottom(t)(h)` is solely used in `\pcol@addmarginpar`¹⁸⁷ to let $B = \pcol@getmparbottom@i$ `\@mparbottom` point the bottom of the marginal note below which a new marginal note m , `\pcol@getmpbelt` whose natural top is at t and occupancy height is h , in the current column c and the page p is placed, or 0 if the side margin has had no marginal notes yet. It is also `\defines` $M' = \pcol@mpblist$ having $mpar(t', t'+h)$ for the new marginal note m placed at t' in it and being replaced with one of $M_{\{L,R\}}^{\{l,r\}}$ in $\pi^m(p)$ by `\pcol@setmpbelt`.

First we examine if $\pi^m(p) = \pcol@mparbottom$ is empty and if so we simply let $B = 0$ and $M' = (mpar(t, t+h))$ because there are no marginal notes in the page p at all. Otherwise we obtain $M \in \{M_X^x \mid X \in \{L, R\}, x \in \{l, r\}\}$ in `\reserved@a` according to the side margin to which the new marginal note m goes to, i.e., according to `\if@firstcolumn` and $c < C_L$ or not, by `\pcol@getmparbottom@i` giving it the body of $\pi^m(p)$ by `\expandafter`. Then we apply `\pcol@getmpbelt` to each $mpar(t_i, b_i) \in M$ to have $t_k = \min\{t_i \mid t_i \geq t, t_i - \max(t, b_{i-1}) \geq h\}$ and let $B = b_{k-1}$ and

$$M' = (mpar(t_1, b_1), \dots, mpar(t_{k-1}, b_{k-1}), mpar(\max(t, B), \max(t, B) + h), mpar(t_k, b_k), \dots, mpar(t_n, b_n))$$

where $n = |M|$, assuming $b_0 = 0$. That is, we try to find the first available gap between marginal notes below t to accommodate the marginal note m of h tall. Then if such t_k is not found because $t > t_n$ to mean m appears below the last marginal note as in natural cases, or $t_i - \max(t, b_{i-1}) < h$ for all t_i s.t. $t_i > t$ to mean there are no available gaps, we let $B = t_n$ and $M' = (M, mpar(\max(t, B), \max(t, B) + h))$ to place m at the bottom.

```

1326 \def\pcol@getmparbottom#1#2{%
1327   \global\@mparbottom\z@
1328   \ifx\pcol@mparbottom\empty
1329     \begingroup
1330     \@tempdimc#2\relax \advance\@tempdimc#1\relax \let\@elt\relax
1331     \xdef\pcol@mpblist{\@elt{\number#1}{\number\@tempdimc}}%
1332   \endgroup
1333 \else
1334   \expandafter\pcol@getmparbottom@i\pcol@mparbottom
1335   \begingroup
1336   \@tempdima#1\relax \@tempdimb#2\relax \@tempswafalse
1337   \let\@elt\pcol@getmpbelt \global\let\pcol@mpblist\empty \reserved@a
1338   \if@tempswa\else
1339     \ifdim\@tempdima<\@mparbottom \@tempdima\@mparbottom \fi
1340     \advance\@tempdimb\@tempdima
1341     \@cons\pcol@mpblist{\@elt{\number\@tempdima}{\number\@tempdimb}}%
1342   \fi
1343 \endgroup
1344 \fi}
1345 \def\pcol@getmparbottom@i#1#2#3#4{%
1346   \ifnum\pcol@currcol<\pcol@ncolleft
1347     \if@firstcolumn \def\reserved@a{#1}\else\def\reserved@a{#2}\fi

```

¹⁸⁷Thus giving t and h as arguments is not necessary but we dare to do it.

```

1348 \else
1349 \if@firstcolumn \def\reserved@a{#3}\else\def\reserved@a{#4}\fi
1350 \fi}
1351 \def\pcol@getmpbelt#1#2{%
1352 \ifdim#1sp<\@tempdima
1353 \global\@mparbottom#2sp\relax \@cons\pcol@mpblist{#1}{#2}}%
1354 \ifdim\@tempdima<#2sp\relax \@tempdima#2sp\relax \fi
1355 \else
1356 \@tempdimc\@tempdima \advance\@tempdimc\@tempdimb
1357 \ifdim#1sp<\@tempdimc
1358 \@tempdima#2sp\relax \global\@mparbottom#2sp\relax
1359 \@cons\pcol@mpblist{#1}{#2}}%
1360 \else
1361 \@cons\pcol@mpblist{\number\@tempdima}{\number\@tempdimc}\@elt{#1}{#2}}%
1362 \@tempswatruue
1363 \def\pcol@getmpbelt##1##2{\@cons\pcol@mpblist{##1}{##2}}
1364 \fi
1365 \fi}
1366

```

`\pcol@setmpbelt` The macro `\pcol@setmpbelt{ $\pi^p(q)$ }\langle $\pi^i(q)$ \rangle\langle $\pi^f(q)$ \rangle\{\pi^s(q)\}\{\pi^m(q)\}` is used solely in `\pcol@addmarginpar` and is applied to Π^+ to update an element $M \in \{M_X^x \mid X \in \{L, R\}, x \in \{l, r\}\}$ in $\pi^m(p)$ with $M' = \pcol@mpblist$ given by `\pcol@getmparbottom`. The structure of the macro is similar to `\pcol@setpnoelt` to update $\pi^p(q)$ s.t. $q \geq p$, but in this macro the target of the update is only p . Then for $q = p$, we invoke `\pcol@setmpbelt@i` giving it the body of $\pi^m(p)$ being $\{M_L^l\}\{M_L^r\}\{M_R^l\}\{M_R^r\}$, or $\langle\emptyset\rangle\langle\emptyset\rangle\langle\emptyset\rangle\langle\emptyset\rangle$ if $\pi^m(p) = \emptyset$, to obtain what $\pi^m(p)$ should have in $\pi_{new}^m(p) = \reserved@a$ in which M_X^x is replaced with M' , where $X = L$ or $X = R$ if $c < C_L$ or not respectively, and $x = l$ or $x = r$ if `\if@firstcolumn = true` or not respectively, and then update $\pi^m(p)$ by `\pcol@defcurrpage{\pi^p(p)}\langle\pi^i(p)\rangle\langle\pi^f(p)\rangle\{\pi^s(p)\}\{\pi_{new}^m(p)\}`.

```

1367 \def\pcol@setmpbelt#1#2#3#4#5{%
1368 {\let\@elt\relax \xdef\pcol@pages{\pcol@pages\pcol@currpage}}%
1369 \ifnum\@tempcnta=\z@
1370 \def\reserved@a{#5}%
1371 \ifx\reserved@a\@empty \pcol@setmpbelt@i{}{}{}{} \else \pcol@setmpbelt@i#5\fi
1372 \pcol@defcurrpage{#1}{#2}{#3}{#4}{\reserved@a}%
1373 \else \gdef\pcol@currpage{\@elt{#1}#2#3#4}{#5}}%
1374 \fi
1375 \advance\@tempcnta\m@ne}
1376 \def\pcol@setmpbelt@i#1#2#3#4{%
1377 \ifnum\pcol@currcol<\pcol@ncolleft
1378 \if@firstcolumn \def\reserved@a{\pcol@mpblist}{#2}{#3}{#4}}%
1379 \else \def\reserved@a{#1}{\pcol@mpblist}{#3}{#4}}%
1380 \fi
1381 \else
1382 \if@firstcolumn \def\reserved@a{#1}{#2}{\pcol@mpblist}{#4}}%
1383 \else \def\reserved@a{#1}{#2}{#3}{\pcol@mpblist}}%
1384 \fi
1385 \fi}
1386

```

`\pcol@mparbottom@zero` The macro $\mathcal{M}_0 = \pcol@mparbottom@zero$ is used in `\@outputpage`, `\pcol@getmparbottom@last` and `\pcol@output@end` to give the default value of $\mathcal{M} = \pcol@mparbottom@out$ with $mpar(0, 0)$ for all elements $M \in \{M_X^x \mid X \in \{L, R\}, x \in \{l, r\}\}$ to mean a page has no marginal notes carrying over from the preceding `paracol` environments.

As for \mathcal{M} , besides the top level initialization to make it \mathcal{M}_0 , it is updated in `\pcol@output@end` through macros `\pcol@getmparbottom@last` and `\pcol@bias@mpbout` to have the last element of each $M \in \{M_X^x \mid X \in \{L, R\}, x = \{l, r\}\}$ in $\pi^m(p_t)$ with transformation from coordinates of columns to that of text area, or directly with \mathcal{M}_0 if the last page will not have post-environment stuff. The resulting \mathcal{M} is at first used in `\pcol@output@end` itself through `\pcol@do@mpbout` to let `\@mparbottom` have the value b of $mpar(t, b)$ in M_L^l or M_L^r according to the side margin which marginal notes in post-environment stuff goes to.

Then \mathcal{M} is passed to the next `paracol` environment if it resides in the page where the previous environment also resides, to be referred to by `\pcol@output@start` which also performs `\pcol@do@mpbout` and `\pcol@bias@mpbout` to let $\pi^m(0)$ have the lists according to \mathcal{M} and `\@mparbottom` which can be modified in post-environment stuff of the previous environment or in other word in pre-environment stuff of starting environment. By this setting of $\pi^m(0)$, marginal note placement in the starting page is aware of the marginal notes having been placed in previous environments and in pre-environment stuff and thus can correctly examines if a marginal note to be added in a margin collide the last one in the margin. On the other hand, if the post-environment stuff encounters a page break before a new environment starts, our own `\@outputpage` should be invoked at the page break to let $\mathcal{M} = \mathcal{M}_0$ because the marginal notes in previous environments do not affect those in the new environment.

Note that \mathcal{M} is also referred to and updated by `\pcol@do@mpb@all@i` and `\pcol@do@mpb@all@ii` because they are used in `\pcol@bias@mpbout` and `\pcol@getmparbottom@last` through `\pcol@do@mpb@all`.

```
1387 \gdef\pcol@mparbottom@zero{{\@elt{0}{0}}{\@elt{0}{0}}{\@elt{0}{0}}{\@elt{0}{0}}}
1388 \global\let\pcol@mparbottom@out\pcol@mparbottom@zero
1389
```

`\pcol@do@mpbout` The macro `\pcol@do@mpbout` is used in `\pcol@output@start` and `\pcol@output@end` to perform operations specified by `\pcol@do@mpbout@whole` and `\pcol@do@mpbout@elem`. The `\pcol@do@mpbout@whole` macro just invokes `\pcol@do@mpbout@i` giving it all $M \in \{M_X^x \mid X \in \{L, R\}, x \in \{l, r\}\}$ by `\pcol@do@mpbout@elem` `\expandafter`.

Then `\pcol@do@mpbout@i` determines the side margin $x \in \{l, r\}$ letting $x = l$ iff `\if@mparswitch = true`, $page(p) \bmod 2 = 0$ and `\if@reversemargin = false`, to invoke `\pcol@do@mpbout@whole` giving it all M but M_L^x whose sole element $mpar(t, b)$ may be modified by `\pcol@do@mpbout@elem\@elt{t}{b}`.

In `\pcol@output@start`, `\pcol@do@mpbout@whole` is to `\xdefine` \mathcal{M} with all M and `\pcol@do@mpbout@elem` is expanded to $mpar(0, B)$, where $B = \@mparbottom$, regardless of M_L^x so that it is replaced with $(mpar(0, B))$ in the modified \mathcal{M} keeping other elements unchanged. In `\pcol@output@end`, `\pcol@do@mpbout@whole` is to throw all M away into a `\hbox` while `\pcol@do@mpbout@elem` lets $B = b$ so that the bottom of the last marginal note in the side margin specified by x is passed to post-environment stuff through `\@mparbottom`.

```
1390 \def\pcol@do@mpbout{\expandafter\pcol@do@mpbout@i\pcol@mparbottom@out}
1391 \def\pcol@do@mpbout@i#1#2#3#4{\@tempcnta\@cne
1392 \if@mparswitch \ifodd\c@page\else \@tempcnta\@m@ne \fi\fi
1393 \if@reversemargin \@tempcnta-\@tempcnta \fi
1394 \ifnum\@tempcnta<\z@
1395 \pcol@do@mpbout@whole{\pcol@do@mpbout@elem#1}{#2}{#3}{#4}%
1396 \else
1397 \pcol@do@mpbout@whole{#1}{\pcol@do@mpbout@elem#2}{#3}{#4}%
1398 \fi}
1399
```

`\pcol@bias@mpbout` The macro `\pcol@bias@mpbout{y}` is used in `\pcol@output@start` with $-y$ being the height-plus-depth of pre-environment stuff, in `\pcol@output@end` with y being that of spanning stuff in the last page, and in `\pcol@getmparbottom@last{y}` with its argument y . The macro modifies $mpar(t, b)$ in all $M \in \{M_X^x \mid X \in \{L, R\}, x \in \{l, r\}\}$ of \mathcal{M} so that they have $mpar(t+y, b+y)$ for the transformation from text area coordinates to columns in the first and third, while for the reverse transformation in the second, by invoking `\pcol@do@mpb@all` giving it \mathcal{M} and letting `\reserved@a` have `\pcol@bias@mpbout{i}{y}`. That is, `\pcol@bias@mpbout{i}{y}\@elt{t}{b}\@nil` is then invoked in `\pcol@do@mpb@all@ii` with t and b from $mpar(t, b)$ in each M , and then it `\defines` `\reserved@b` with $mpar(t+y, b+y)$ so that updated M has it.

```

1400 \def\pcol@bias@mpbout#1{\def\reserved@a{\pcol@bias@mpbout{i}{#1}}%
1401 \pcol@do@mpb@all\pcol@mparbottom@out}
1402 \def\pcol@bias@mpbout@i#1\@elt#2#3\@nil{%
1403 \dimen#2sp\relax \advance\dimen#1\relax
1404 \dimen@ii#3sp\relax \advance\dimen@ii#1\relax
1405 \def\reserved@b{\@elt{\number\dimen@}{\number\dimen@ii}}
1406

```

`\pcol@getmparbottom@last` The macro `\pcol@getmparbottom@last{y}` is used solely in `\pcol@output@end` to let $\mathcal{M} = \{m_L^l\}\{m_L^r\}\{m_R^l\}\{m_R^r\}$, where $m_X^x = mpar(t_n, b_n) \in M_X^x$, $n = |M_X^x|$ assuming $mpar(t_0, b_0) = mpar(y, y)$, and y is the negative counterpart of the height-plus-depth of the spanning stuff in the last page. Therefore, \mathcal{M} is let have the occupancy information of the last marginal note if any, or the top edge of text area otherwise, in each margin.

The macro at first examines if $\pi^m(p_t) = \emptyset$ and, if so, lets all elements in \mathcal{M} have $(mpar(y, y))$ by letting it \mathcal{M}_0 and then adding y to each $t = b = 0$ by `\pcol@bias@mpbout` giving it y . Otherwise, i.e., if $\pi^m(p_t) \neq \emptyset$, it invokes `\pcol@do@mpb@all` giving it $\pi^m(p_t)$ and letting `\reserved@a` have `\pcol@getmparbottom@last@i{y}`. That is, `\pcol@getmparbottom@last@i{y}mpar(t_1, b_1) \cdots mpar(t_n, b_n)\@nil` is then invoked in `\pcol@do@mpb@all@ii` for each M_X^x to let `\reserved@b` have $mpar(y, y)$ at first and then to let it have $mpar(t_i, b_i)$ for all $i \in [1, n]$. Therefore, `\reserved@b` should finally have $mpar(t_n, b_n)$ assuming $t_0 = b_0 = y$, and then becomes m_X^x .

```

1407 \def\pcol@getmparbottom@last#1{%
1408 \ifx\pcol@mparbottom\@empty
1409 \global\let\pcol@mparbottom@out\pcol@mparbottom@zero
1410 \pcol@bias@mpbout{#1}%
1411 \else
1412 \def\reserved@a{\pcol@getmparbottom@last@i{#1}}%
1413 \pcol@do@mpb@all\pcol@mparbottom
1414 \fi}
1415 \def\pcol@getmparbottom@last@i#1#2\@nil{%
1416 \edef\reserved@b{\@elt{\number#1}{\number#1}}%
1417 \def\@elt##1##2{\def\reserved@b{\@elt{##1}{##2}}}%
1418 #2\let\@elt\relax}
1419

```

`\pcol@do@mpb@all` The macro `\pcol@do@mpb@all<L>` is used in `\pcol@bias@mpbout` with $L = \mathcal{M}$ and in `\pcol@do@mpb@all@i` `\pcol@getmparbottom@last` with $L = \pi^m(p_t)$, to process all four elements M_X^x in L applying `\reserved@a` to each of them and to let \mathcal{M} have the result through `\reserved@b`. The macro simply invokes `\pcol@do@mpb@all@i` giving it the body of L by `\expandafter`. Then `\pcol@do@mpb@all@i\{M_L^l\}\{M_L^r\}\{M_R^l\}\{M_R^r\}` initialize $\mathcal{M} = \emptyset$ and then invokes `\pcol@do@mpb@all@ii` four times giving it each M_X^x . Then `\pcol@do@mpb@all@ii` $mpar(t_1, b_1) \cdots$

$\mathit{mpar}(t_n, b_n)\@nil$ invokes `\reserved@a` giving it all of $\mathit{mpar}(t_i, b_1)$ at once to process them and to have the result in `\reserved@b` being added to \mathcal{M} .

```

1420 \def\pcol@do@mpb@all#1{\expandafter\pcol@do@mpb@all@i#1}
1421 \def\pcol@do@mpb@all@i#1#2#3#4{\begingroup \let\@elt\relax
1422   \gdef\pcol@mparbottom@out{}}%
1423   \pcol@do@mpb@all@ii#1\@nil\pcol@do@mpb@all@ii#2\@nil
1424   \pcol@do@mpb@all@ii#3\@nil\pcol@do@mpb@all@ii#4\@nil
1425   \endgroup}
1426 \def\pcol@do@mpb@all@ii#1\@nil{%
1427   \reserved@a#1\@nil
1428   \xdef\pcol@mparbottom@out{\pcol@mparbottom@out{\reserved@b}}
1429

```

11.7 Synchronization

`\pcol@sync` The macro `\pcol@sync` is invoked solely from `\pcol@output@switch` for explicit synchronization in the following three cases in which `\ifpcol@sync ∨ \ifpcol@clear = true` commonly.

- `\ifpcol@sync ∧ ¬\ifpcol@clear` to mean ordinary synchronized column-switching.
- `\ifpcol@sync ∧ \ifpcol@clear` to mean pre-flushing column height check.
- `¬\ifpcol@sync ∧ \ifpcol@clear` to mean page flushing.

In any cases¹⁸⁸, first we invoke `\pcol@flushcolumn` for all $c \in [0, C)$ to flush the current column-page of c into S_c if the column-page is not in p_t , i.e., $\kappa_c(\beta^p) < p_t$ and then, if we have deferred floats, to ship out following float pages up to $p_t - 1$ into S_c and to place them in p_t . This float placement in p_t is only for top and bottom floats in synchronized column-switching, while a float column may be made in other cases. Then we ship out all pages p such that $p < p_t$ by `\pcol@outputcolumns` giving argument 1. After that, we obtain the page context of p_t by `\pcol@getcurrpinfo`.

Next, we measure the vertical sizes of the contents in the current column-page of c which is now in p_t for all $c \in [0, C)$ by `\pcol@measurecolumn` as follows, where $h(x)$ and $d(x)$ are the height and depth of an object x respectively.

$$\begin{aligned}
\sigma &= \text{\floatsep} & \sigma_t &= \begin{cases} \kappa_c(\xi) & \kappa_c(\xi) < \infty \\ \text{\textfloatsep} & \kappa_c(\xi) = \infty \end{cases} & \sigma_b &= \text{\textfloatsep} \\
f_c(t) &= (\kappa_c(\lambda_t) \neq ()) & f_c(m) &= (\kappa_c(\beta) \neq \text{\vbox{}}) \\
f_c(f) &= (\kappa_c(\tau) \neq \perp) & f_c(b) &= f_c(b') = (\kappa_c(\lambda_b) \neq ()) \\
F_c(X) &= \exists x \in X : f_c(x) & f_b &= \text{\ifpcol@bfbottom} \\
v_c(t) &= \text{\skip} \cdot \kappa_c(\beta) = \sum_{\phi \in \kappa_c(\lambda_t)} (h(\phi) + d(\phi)) + (|\kappa_c(\lambda_t)| - 1) \cdot \sigma + \sigma_t \\
v_c(m) &= h(\kappa_c(\beta)) + d(\kappa_c(\beta)) \\
v_c(f) &= h(\kappa_c(\tau)) + d(\kappa_c(\tau)) \\
v_c(b) &= \sum_{\phi \in \kappa_c(\lambda_b)} (h(\phi) + d(\phi)) + (|\kappa_c(\lambda_b)| - 1) \cdot \sigma + \sigma_b \\
v_c(b') &= v_c(b) + \sigma_b
\end{aligned}$$

¹⁸⁸In the last case of page flushing, invoking `\pcol@flushcolumn` is redundant because it is made $p = p_t$ by pre-flushing column height check always preceding the flushing, but the invocation is harmless.

$$\begin{aligned}
\delta_c &= \begin{cases} \kappa_c(\delta) & f_c(m) \\ \infty & \neg f_c(m) \end{cases} \\
size_c(x) &= \begin{cases} v_c(x) & f_c(x) \\ 0 & \neg f_c(x) \end{cases} \\
SIZE_c(X) &= \begin{cases} \sum_{x \in X} size_c(x) & F_c(X) \\ -\infty & \neg F_c(X) \end{cases} \\
V_T = \backslash@tempdima &= \max_{0 \leq c < C} \{SIZE_c(\{t, m\})\} \\
V_B = \backslash@tempdimb &= \max_{0 \leq c < C} \{size_c(f) + size_c(b)\} \\
V_P = \backslash@pageht &= \max_{0 \leq c < C} \{SIZE_c(\{t, m, f, b\})\} \\
V'_P = \backslashpcol@colht &= \max_{0 \leq c < C} \{SIZE_c(\{t, m, f, b'\})\} \\
D_T = \backslash@tempdimc &= \min\{\delta_c \mid SIZE_c\{t, m\} = V_T\} \\
d_c &= \begin{cases} 0 & f_c(b) \wedge (\neg f_c(f) \vee f_b) \\ d(\kappa_c(\tau)) & f_c(f) \wedge (\neg f_c(b) \vee \neg f_b) \\ d(\kappa_c(\delta)) & \neg f_c(b) \wedge \neg f_c(f) \end{cases} \\
D_P = \backslash@pagedp &= \min\{d_c \mid SIZE_c(\{t, m, f, b'\}) = V'_P\} \\
c_{\max} = \backslash@tempcntb &= \arg \max_{0 \leq c < C} \{SIZE_c(\{t, m, f, b\})\}
\end{aligned}$$

That is, V_T is the maximum of combined vertical size (height plus depth) of the top floats and the main vertical list, V_B is that of the footnotes and bottom floats, and V_P is that of all items. V'_P is similar to V_P but we add `\textfloatsep` to the size of bottom floats. Note that V_T , V_P and V'_P are $-\infty$ if any column-pages don't have corresponding items, while $V_B = 0$ if so. Also note that c_{\max} is the ordinal of the column whose size is V_P .

D_T and D_P are the minimum δ_c and d_c respectively among those gives V_T and V'_P respectively, where δ_c is $\kappa_c(\delta)$ if $f_m = true$ or ∞ otherwise, and d_c is 0 if c has bottom float, or the depth of the last footnote if any and without any bottom float, or $\kappa_c(\delta)$ otherwise. The reason why D_T and D_P have minimums is that they are set into `\prevdepth` for the items just following the synchronization point, and thus a smaller value results in a larger interline skip and the special value `-1000pt` to inhibit the skip by, e.g., `\nointerlineskip`, is given the highest priority.

Note that V'_P and D_P are only for the last page and thus referred to by `\pcol@output@end` to close the environment, and the former is done by `\pcol@makeflushedpage` if it works on the page. The reason why we add `\textfloatsep` to V'_P is to make the last page well separated from the post-environment stuff if the tallest column, taking the addition into account, has bottom floats. Also note that we let `\ifpcol@dfloats = false` before scanning columns with `\pcol@measurecolumn` so that the switch becomes `true` after the scan iff a column has deferred floats (in the last page).

```

1430 %% Special Output Routines: Synchronization
1431
1432 \def\pcol@sync{%
1433   \pcol@currcol\z@ \@whilenum\pcol@currcol<\pcol@ncol\do\pcol@flushcolumn
1434   \pcol@outputcolumns\@ne
1435   \pcol@getcurrrpinfo{\global\c@page}{\global\@colht}{\global\topskip}%
1436   \@tempdima-\maxdimen \@tempdimb-\maxdimen \pcol@colht-\maxdimen
1437   \@pageht-\maxdimen \@tempdimc\maxdimen \@pagedp\maxdimen \@tempcntb\z@
1438   \pcol@dfloatsfalse

```

As described above, any items can be empty, naturally for top floats, footnotes and bottom floats, but also including main vertical lists if the current column-pages were not in p_t before the invocation of `\pcol@flushcolumn`. Moreover, all main vertical lists can be empty if all leading column-pages just have started (by `\newpage`, for example). More weirdly, the case of all-empty main vertical lists can be accompanied with other non-empty items when columns have floats or footnotes which cannot be in $p_t - 1$ but are found their places in p_t .

Taking it into account that any items can be empty and the other item of page-wise footnotes, we have to determine whether the following two operations are taken; invocation of `\pcol@synccolumn` for each column-page to set a synchronization point or to add an infinite stretch (and shrink) to its bottom; and the examination of the height of each column-page, taking the synchronization point to be set into account, to tell the necessity of explicit page break with `\ifpcol@flush = true`. For the latter we calculate the examination target $V = \text{\@tempdimb}$ to be compared with $\pi^h(p_t)$, while for the former we determine the value of a switch $f = \text{\if@tempswa}$ so that we invoke `\pcol@synccolumn` iff $f = true$ and $V \geq 0$.

For synchronized column-switching with `\ifpcol@clear = false`, we let $f = (V_T \geq 0)$ to mean at least one column-page has a top float or non-empty main vertical list, i.e., $F_c(\{t, m\}) = true$ for some $c \in [0, C)$. That is if $V_T < 0$, since the next items added to all column-pages are placed at the top of the page¹⁸⁹, we don't need to set synchronization points in them. As for V , we let $V = \max(V_T, 0) + \max(V_B, 0) + v^f$ where v^f is the sum of height-plus-depth of $\pi^f(p_t)$ and `\skip\pi^f(p_t)` if p_t has page-wise footnotes, or 0 otherwise¹⁹⁰. Note that it can be $V_P + v^f \leq \pi^h(p_t) < V_T + V_B + v^f = V$ to mean setting the synchronization point at V_T below p_t 's top edge would push bottom stuff beyond its bottom edge and thus we need an explicit page break to place the point at the top of $p_t + 1$ (in usual cases).

For pre-flushing column height check or page flushing with `\ifpcol@clear = true` on the other hand, we let $f = \neg \text{\pcol@sync}$ to invoke `\pcol@synccolumn` only for page flushing and thus not for pre-flushing column height check. As for V , we let $V' = V'_P$ or $V' = V_P$ according as we working on last page or not, and then let $V = \max(V', 0) + v^f$ or $V = V'$ according as p_t has page-wise footnotes or not. That is, we have to invoke `\pcol@synccolumn` unless p_t is perfectly empty.

Then if `\ifpcol@clear = false` and $\max(V, V - D'_T + V_E) > \pi^h(p_t)$ where $D'_T = D_T$ if $0 \leq D_T < \infty$ or 0 otherwise, or `\ifpcol@clear = true` and $V > \pi^h(p_t)$ ¹⁹¹, we flush the page. That is, if the condition above holds, we let `\ifpcol@flush = true` and $d = \text{\pcol@nextcol} = c_{\max}$ to tell `\pcol@switchcol` or `\pcol@flushclear` to make an explicit page break in the column c_{\max} from which we restart, and $f = false$ to skip `\pcol@synccolumn` to postpone the explicit synchronization. Note that the bias $V_E = \text{\pcol@ensurevspace}$ in synchronized column-switching is to avoid breaking a column-page just below the synchronization point due to too small space below the point, less than `\baselineskip` in default but can be other threshold explicitly defined by `\ensurevspace`. That is, since $V - D_T + k \text{\baselineskip}$ usually means the vertical position at which k -th baseline below the synchronization point is placed, the flushing condition with $V_E = k \text{\baselineskip}$ ensures that the page is flushed iff the space below the point cannot accommodate k lines. Also note that necessary flushing with

¹⁸⁹In usual cases, but it can mean some of them have negative vertical sizes. Even though we can detect such a very unlikely special case, it is very tough to define the reasonable synchronization point above the top of p_t . Therefore, we assume the point is at the top of p_t and thus do nothing.

¹⁹⁰In the real implementation, $V = -\infty$ if $V_T = V_B = -\infty$ and no page-wise footnotes are presented, but this difference does not affect the decisions because $f \wedge (V \geq 0) = (V_T < 0) \wedge (V \geq 0) = false$ and $V \leq \pi^f(p_t)$ with either $V = 0$ or $V = -\infty$.

¹⁹¹The examination is redundant in page flushing with `\ifpcol@sync = false` because it is assured that no overflow happens in any column-page by pre-flushing column height check and explicit page breaking, but is not harmful.

$V > \pi^h(p)$ assuredly takes place even when D_T is unusually large and/or V_E is negative to make $-D_T + V_E < 0$.

Finally if $V \geq 0$ and $f = true$, we invoke `\pcol@synccolumn` for each column $c \in [0, C)$ to set a synchronization point in it or to add an infinite stretch (and shrink) at its bottom for flushing.

```

1440 \@tempwatrue \global\pcol@flushfalse
1441 \ifpcol@clear
1442   \ifpcol@lastpage \@tempdimb\pcol@colht \else \@tempdimb\@pageht \fi
1443   \ifpcol@sync \@tempwafalse \fi
1444   \else
1445     \ifdim\@tempdima<\z@ \@tempwafalse
1446     \else\ifdim\@tempdimb<\z@ \@tempdimb\@tempdima
1447     \else \advance\@tempdimb\@tempdima
1448     \fi\fi
1449   \fi
1450 \ifpcol@scfnote\ifvoid\pcol@footins\else
1451   \ifdim\@tempdimb<\z@ \@tempdimb\z@ \fi
1452   \advance\@tempdimb\ht\pcol@footins \advance\@tempdimb\dp\pcol@footins
1453   \advance\@tempdimb\skip\pcol@footins
1454 \fi\fi
1455 \dimen@\@tempdimb
1456 \ifpcol@clear\else \ifdim\dimen@<\z@\else
1457   \ifdim\@tempdimc=\maxdimen\else \ifdim\@tempdimc<\z@\else
1458     \advance\dimen@-\@tempdimc
1459   \fi\fi
1460   \advance\dimen@\pcol@@ensurevspace
1461   \ifdim\dimen@<\@tempdimb \dimen@\@tempdimb \fi
1462 \fi\fi
1463 \ifdim\dimen@>\@colht
1464   \global\pcol@flushtrue \@tempwafalse \pcol@nextcol\@tempcntb
1465 \fi
1466 \ifdim\@tempdimb<\z@\else \if@tempwa
1467   \pcol@currcol\z@ \@whilenum\pcol@currcol<\pcol@ncol\do\pcol@synccolumn
1468 \fi\fi}
1469

```

`\pcol@flushcolumn` The macro `\pcol@flushcolumn` is invoked for each column $c \in [0, C)$ from `\pcol@sync` to ship out the current column-page of c into S_c if it is not leading one, i.e., $p = \kappa_c(\beta^p) < p_t$. The macro also ships out float pages from $p+1$ up to p_t-1 if we have deferred floats to fill them and, if this float flushing still leaves deferred floats, puts some of them to the leading column-page being current now as its top and/or bottom floats.

First we obtain the column-context in κ_c by `\pcol@getcurrcol` and examines if $p = \kappa_c(\beta^p) < p_t$. If it does not hold to mean c has leading column-page, we do nothing.

Otherwise, we save `\ifpcol@lastpage` into `\ifpcol@lastpagesave` turning the former switch *false* because we are working on a non-leading column-page definitely in a non-last page. Then we put the contents of the current column-page $\kappa_c(\beta^b)$ adding `\vfil` at its tail into `\box255` being the T_EX's standard interface to carry the main vertical list for `\output` routine. We also move everything in $\kappa_c(\tau)$ obtained by `\pcol@getcurrfoot` into `\footins` and return $\kappa_c(\tau)$ to `\@freelist` if $\kappa_c(\tau^b)$ is not void.

Then we obtain p 's page context by `\pcol@getcurrpage` and, if it has page-wise footnotes in $\pi^f(p)$, we shrink `\@colht` by the space required for the footnotes using `\pcol@shrinkcolbyfn` and add the stretch/shrink components of `\skip \cdot \pi^f(p)` at the bottom of `\box255` by `\pcol@`

`\unvbox\@cclv`, as we did in `\pcol\@makecol`. Otherwise we take a special care of the case that the height-plus-depth of $\kappa_c(\beta^b)$ is greater than $\pi^h(p)$ due to that its height is almost equal to $\pi^h(p)$ and thus its depth makes the height-plus-depth exceeding $\pi^h(p)$. This excess is revealed by the `\vfil` we just have added making the height-plus-depth the height of `\box255`, and would cause overfull in `\@makecol` and `\pcol\@makecol` because they need the height, i.e., not height-plus-depth, of `\box255` not exceeding $\pi^h(p)$. Therefore if it happens, we remove the `\vfil` and cap the height of `\box255` by `\@maxdepth` to pretend as if the box is directly passed from TeX's page builder.

Next we examine if $\kappa_c(\rho_t)$ was made ∞ by `\pcol\@makefcolumn` invoked from this macro itself when it processed the column-page in the previous pre-flushing column height check for environment closing, which found a page break should be done. That is, $\kappa_c(\rho_t) = \infty$ means the current column-page was once judged to be in the last page but pre-flushing column height check forced a page break to make it non-last, it should have all deferred floats now listed in $\kappa_c(\lambda_t)$ at `\end\@paracol`, but their total size is less than the threshold to make a usual float column for the last page. If so, since the page is not last now, we put all floats in $\kappa_c(\lambda_t)$ by `\pcol\@makefcolpage` as the ship-out image in `\@outputbox`, ignoring the contents added to `\box255` in the operations above because $\kappa_c(\beta^b)$ should be empty, and letting $\kappa_c(\rho_t) = 0$ to mean the floats have been processed¹⁹².

Otherwise, since the column-page can be put as usual, we invoke `\pcol\@makecol`¹⁹³ giving `\@maxdepth` to it to build the complete column-page in `\@outputbox` with depth capping and with the following setting¹⁹⁴.

$$\begin{aligned} \box255 &= \kappa_c(\beta^b) & \footins &= \kappa_c(\tau) & \colht &= \pi^h(\kappa_c(\beta^p)) \\ \midlist &= \kappa_c(\lambda_m) & \toplist &= \kappa_c(\lambda_t) & \botlist &= \kappa_c(\lambda_b) \end{aligned}$$

Finally, regardless of $\kappa_c(\rho_t) = \infty$ or not, the resulting `\@outputbox` becomes $s_c(p)$ and is added to the tail of S_c by `\@cons`.

```

1470 \def\pcol\@flushcolumn{%
1471   \pcol\@getcurrcol
1472   \ifnum\count\@currbox<\pcol\@toppage
1473     \ifpcol\@lastpage \pcol\@lastpagesavetrue \else \pcol\@lastpagesavefalse \fi
1474     \pcol\@lastpagefalse
1475     \pcol\@page\count\@currbox
1476     \setbox\@cclv\ vbox{\unvbox\@currbox \vfil}%
1477     \ifvoid\pcol\@currfoot\else
1478       \pcol\@Fb
1479       \@cons\@freelist\pcol\@currfoot
1480       \pcol\@Fe\@flushcolumn(colfn)}%
1481     \fi
1482     \pcol\@getcurrfoot\box
1483     \pcol\@getcurrpage
1484     \ifvoid\pcol\@footins
1485       \ifdim\ht\@cclv>\colht
1486         \setbox\@cclv\ vbox{\boxmaxdepth\@maxdepth \unvbox\@cclv \unskip}%
1487       \fi
1488     \else
1489       \pcol\@shrinkcolbyfn\colht\pcol\@footins\relax

```

¹⁹² $\kappa_c(\lambda_t)$ can have any values other than ∞ because definitely it will not be referred to in its inherent sense in the situation with no further float additions and no deferred floats.

¹⁹³Neither `\pcol\@makecol` because `\box255` has `\vfil` at its tail and the column-page should be short enough, nor `\@makecol` because we need to ensure the depth of resulting `\@outputbox` is capped.

¹⁹⁴L^AT_EX's has another `\insert` named `\@kludgeins` for `\enlargepage` but `\paracol` does not cares about it.

```

1490     \setbox\@cclv\vbox{\pcol@unvbox@cclv\pcol@footins}%
1491     \fi
1492     \pcol@Logstart{\pcol@flushcolumn(\number\c@page:\number\pcol@currcol)}
1493     \ifdim\@toproom=\maxdimen
1494         \setbox\@outputbox\pcol@makefcolpage \global\@toproom\z@
1495     \else
1496         \pcol@@makecol\@maxdepth
1497     \fi
1498     \pcol@Logend\pcol@flushcolumn
1499     \global\setbox\@currbox\box\@outputbox
1500     \expandafter\@cons\csname pcol@shipped\number\pcol@currcol@endcsname
1501     \@currbox

```

Then for each $q \in [p+1, p_t-1]$, we repeat the followings; get q 's page context by `\pcol@getcurrpage`; shrink `\@colht` by `\pcol@shrinkcolbyfn` if q has page-wise footnotes; try to make a float column in `\@outputbox` by `\@makefcolumn` giving it `\@deferlist` being $\kappa_c(\lambda_d)$ at initial but will be shrunk; if the float column is made, acquire an `\insert` by `\@next` to keep the contents of `\@outputbox` and to be added to the tail of S_c by `\@cons`.

```

1502     \advance\pcol@page\@ne
1503     \ifx\@deferlist\@empty\else
1504         \@whilenum\pcol@page<\pcol@toppage\do{%
1505             \pcol@getcurrpage
1506             \ifvoid\pcol@footins\else
1507                 \pcol@shrinkcolbyfn\@colht\pcol@footins\relax
1508             \fi
1509             \@makefcolumn\@deferlist
1510             \if@fcolmade
1511                 \pcol@Fb
1512                 \@next\@currbox\@freelist{\global\setbox\@currbox\box\@outputbox}%
1513                 \pcol@ovf
1514                 \pcol@Fe{flushcolumn(fcol)}%
1515                 \expandafter\@cons
1516                 \csname pcol@shipped\number\pcol@currcol@endcsname\@currbox
1517             \fi
1518             \advance\pcol@page\@ne}%
1519     \fi

```

Next, since we reach p_t and thus restore `\ifpcol@lastpage` from `\ifpcol@lastpagesave` because the top page can be last. Then we acquire the current column-page of c which is now in p_t and thus empty, by `\@next`. Then we let `\@colht = \@colroom = $\pi^h(p_t)$` by `\pcol@getcurrpinf` but shrinking them by `\pcol@shrinkcolbyfn` if p_t has page-wise footnotes, and reinitialize the float placement parameters by `\pcol@floatplacement`. Then, if $\kappa_c(\lambda_d)$ still has some floats, we make a float column for some of them in the top page by `\pcol@makefcolumn` if `\ifpcol@clear = true` meaning flushing, or try to move some of them to $\kappa_c(\lambda_t) = \@toplist$ and/or $\kappa_c(\lambda_b) = \@botlist$ by `\pcol@trynextcolumn` otherwise. Note that since `\@colroom` is used in `\pcol@makefcolumn` as a working register, we let `\@colroom = $\pi^h(p_t)$` again after its invocation. After that we save column-context including those given by `\pcol@floatplacement` and modified by `\pcol@makefcolumn` or `\pcol@trynextcolumn` into κ_c by `\pcol@setcurrcolnf` because all footnotes are shipped out, and let $\kappa_c(\beta^p) = p_t$. We also let $\kappa_c(\beta^r) = \@colroom$ possibly modified by `\pcol@trynextcolumn` but after canceling the shrinkage of `\@colht` due to page-wise footnotes, i.e., $\kappa_c(\beta^r) \leftarrow \kappa_c(\beta^r) + (H - H')$ where H and H' are `\@colht` before and after the shrinkage respectively.

```

1520     \ifpcol@lastpagesave \pcol@lastpagetrue \fi

```

```

1521 \pcol@Fb
1522 \@next\@currbox\@freelist{\global\setbox\@currbox\ vbox{}}\pcol@ovf
1523 \pcol@Fe{flushcolumn(col)}%
1524 \pcol@getcurrpinfo\@tempcnta{\global\@colht}\@tempskipa
1525 \@pageht\@colht
1526 \ifvoid\pcol@footins\else \pcol@shrinkcolbyfn\@colht\pcol@footins\relax \fi
1527 \global\@colroom\@colht \pcol@floatplacement
1528 \ifx\@deferlist\@empty\else
1529   \ifpcol@clear
1530     \pcol@makefcolumn \global\@colroom\@colht
1531   \else
1532     \pcol@trynextcolumn
1533   \fi\fi
1534 \pcol@setcurrcolnf
1535 \global\count\@currbox\pcol@toppage
1536 \advance\@pageht-\@colht \advance\@pageht\@colroom
1537 \global\dimen\@currbox\@pageht
1538 \fi %\ifnum\count\@currbox<\pcol@toppage
1539 \advance\pcol@currcol\@ne}
1540

```

`\pcol@makefcolumn` The macro `\pcol@makefcolumn` is invoked solely from `\pcol@flushcolumn` to put deferred floats in the currently empty column-page of c in the top page p_t which is being flushed. Since we have to take special care of the case of environment closing, we cannot do this operation by `\@makefcolumn`, while in other cases for `\flushpage` and `\clearpage` we also have to pay a small attention.

First, we scan the copy of $\kappa_c(\lambda_d)$ applying `\pcol@makefcolelt` to each element to have the floats to be put in `\@toplist`, which is assuredly empty because the current column-page of c has already been shipped out to empty it, and those still deferred in $\kappa_c(\lambda_d)$. Prior to scan, we let $H_r = \pi^h(p_t)^{195} + \alpha$, as the space initially available for floats each of which being ϕ is assumed to consume $v(\phi) = h(\phi) + d(\phi) + \alpha$, and $H_t = \@colroom = -\alpha$ as the initial value of the accumulated size of $v(\phi)$ for all ϕ to be put, where $\alpha = \floatsep$ if `\ifpcol@lastpage = true` as discussed afterward, or $\alpha = \@fpsep$ otherwise.

Then if the resulting `\@toplist` is not empty¹⁹⁶, we examine if p_t is the last page (`\ifpcol@lastpage = true`) and $\kappa_c(\lambda_d)$ is empty to mean the last column-page have all deferred floats. This case is subtle because if we make the column-page a float column it can be sparse and unnecessarily throw the post-environment stuff to the next page. Therefore, we intend to pack floats to the page top as top floats and thus have let $\alpha = \floatsep$ in the building process of `\@toplist` above¹⁹⁷, but it may make a too tall column-page only having floats shrinking the post-environment stuff in the page. In addition, if other columns have float pages in the last page, packing the floats as top floats should give inconsistent appearance but we don't know whether the columns following c has float pages. Therefore, we performs this float packing if making the float page gives too sparse, more specifically if $H_t < \floatpagefraction \times \pi^h(p_t) = \@fpmin$, but postpone the final decision until the column-page is eventually shipped out by `\pcol@flushcolumn` or `\pcol@makeflushedpage`. That is, if the all conditions above hold, we keep `\@toplist` so that it is saved in $\kappa_c(\lambda_t)$ and let $\kappa_c(\rho_t) = \infty$ to indicate that the column has pending floats. Note that the floats are shipped out by `\pcol@flushcolumn` if the pre-flushing column height check currently being performed finds a too tall column in p_t to

¹⁹⁵ $\pi^h(p_t)$ referred in this macro may have been shrunk by page-wise footnotes in `\pcol@flushcolumn`.

¹⁹⁶It can happen if the first float is larger than $\pi^h(p_t)$.

¹⁹⁷We dare to do it knowing the natural component of `\floatsep` is a little bit (4pt) larger than that `\@fpsep` and the possibility of having fewer floats than those given by `\@makefcolumn`.

force a page break making p_t non-last. Also note that, in any cases, letting $\kappa_c(\rho_t) = \infty$ is safe because no longer we will have any float additions to the column-page.

Otherwise, i.e., if we are working on a non-last page to be flushed or a float column is to be made for the last page, we put all floats in `\@toplist` in a `\vbox` of $\pi^h(p_t)$ tall by `\pcol@makefcolpage` and then let $\kappa_c(\beta^b)$ be another `\vbox` having it. This encapsulation of the float column is necessary because $\kappa_c(\beta^b)$ can be put back to the main vertical list after the pre-flushing column height check to remove skips above and below the floats, namely `\@fptop` and `\@fpbot`, if the contents were not encapsulated.

```

1541 \def\pcol@makefcolumn{%
1542   \ifpcol@lastpage \@tempdimc\floatsep \else \@tempdimc\@fpsep \fi
1543   \@tempdima\@colht \advance\@tempdima\@tempdimc \global\@colroom<\@tempdimc
1544   \begingroup
1545     \let\@elt\pcol@makefcolelt
1546     \let\reserved@b\@deferlist
1547     \global\let\@deferlist\@empty
1548     \reserved@b
1549   \endgroup
1550   \ifx\@toplist\@empty\else
1551     \@tempswatrue
1552     \ifpcol@lastpage \ifx\@deferlist\@empty \ifdim\@colroom<\@fpmin
1553       \@tempswafalse \global\@toproom\maxdimen
1554     \fi\fi\fi
1555     \if@tempswa \global\setbox\@currbox\vbox{\pcol@makefcolpage}\fi
1556   \fi}

```

`\pcol@makefcolelt` The macro `\pcol@makefcolelt` $\langle\phi\rangle$ is invoked solely from `\pcol@makefcolumn` to be applied to each float element ϕ in (the copy of) $\kappa_c(\lambda_d)$. We examine if $v(\phi) = h(\phi) + d(\phi) + \alpha \leq H_r$ to mean the float column being built has room large enough for the float ϕ . If so, we add ϕ to `\@toplist` by `\@cons`, and let $H_r \leftarrow H_r - v(\phi)$ and $H_t \leftarrow H_t + v(\phi)$. Otherwise, we add ϕ to $\kappa_c(\lambda_d)$ by `\@cons` to make it deferred again, and let $H_r = -\infty$ so that the examinations for any succeeding elements fail.

```

1557 \def\pcol@makefcolelt#1{%
1558   \@tempdimb\ht#1\advance\@tempdimb\dp#1\advance\@tempdimb\@tempdimc
1559   \ifdim\@tempdimb>\@tempdima \@cons\@deferlist#1\relax
1560   \@tempdima-\maxdimen
1561   \else \@cons\@toplist#1\relax
1562   \advance\@tempdima-\@tempdimb \global\advance\@colroom\@tempdimb
1563   \fi}

```

`\pcol@makefcolpage` The macro `\pcol@makefcolpage` is invoked from `\pcol@flushcolumn`, `\pcol@makefcolumn` and `\pcol@imakeflushedpage` to build a float column having floats in `\@toplist`, which is then returned to `\@freelist` and then emptied. The floats are put in a `\vbox` of `\@colht` tall with vertical skips of `\@fptop`, `\@fpsep` and `\@fpbot` above, between and below them respectively. The box is then let be $s_c(p)$ or $\kappa_c(\beta^b)$ explicitly or implicitly by the invokers, with an encapsulation in the case of `\pcol@makefcolumn`.

```

1564 \def\pcol@makefcolpage{\vbox to\@colht{%
1565   \vskip\@fptop \vskip-\@fpsep
1566   \def\@elt##1{\vskip\@fpsep\box##1}\@toplist \vskip\@fpbot}%
1567   \pcol@Fb
1568   \xdef\@freelist{\@freelist\@toplist}\global\let\@toplist\@empty
1569   \pcol@Fe{makefcolpage}%
1570 }

```


`\pcol@measurecolumn` The macro `\pcol@measurecolumn` is invoked for each column $c \in [0, C)$ from `\pcol@sync` to measure the sizes of the top floats $size_c(t)$, main vertical list $size_c(m)$, footnotes $size_c(f)$ and bottom floats $size_c(b)$ in the current column-page in the page p_t . After obtaining the column-context in κ_c by `\pcol@getcurrcol`, we calculate $size_c(t) = \text{skip} \cdot \kappa_c(\beta)$ by `\pcol@addflhd` giving it $\kappa_c(\lambda_t)$ and $\kappa_c(\xi) = \text{pcol@textfloatsep}$ as its arguments also to have `\if@tempswa = f_c(t)`. Note that $\kappa_c(\xi) = \infty$ means the column-page does not have any synchronization points yet and thus `\textfloatsep` is used in the calculation as the skip between the top floats and main vertical list, while the value itself possibly for a MVL-float discussed later is used if $\kappa_c(\xi) < \infty$. We also calculate $size_c(m) = h(\kappa_c(\beta)) + d(\kappa_c(\beta))$, and then the sum of it and $size_c(t)$.

```

1572 \def\pcol@measurecolumn{%
1573   \pcol@getcurrcol
1574   \@tempswafalse
1575   \dimen@z@ \pcol@addflhd\@toplist\pcol@textfloatsep
1576   \global\skip\@currbox\dimen@
1577   \advance\dimen@\ht\@currbox \advance\dimen@\dp\@currbox \dimen@ii\dimen@

```

Next we examine if the main vertical list $\kappa_c(\beta^b)$ is empty by `\pcol@ifempty` and, if so, we let $\delta_c = \kappa_c(\delta) = \infty$ and save it (together with others) by `\pcol@setcurrcol` so that, if the column defines the V_T finally by its top floats, D_T is let ∞ and the fact that the column has empty list is remembered. Otherwise, we let $\delta_c = \kappa_c(\delta)$ and `\if@tempswa = true` to represent $F_c(\{t, m\}) = f_t \vee f_m$ because $f_m = true$. Then we invoke `\pcol@measureupdate` to let $V_T = size_c(t) + size_c(m) = SIZE_c(\{t, m\})$ and $D_T = \delta_c$ if $F_c(\{t, m\}) = true$ and $V_T < SIZE_c(\{t, m\})$.

```

1578   \pcol@ifempty\@currbox
1579   {\global\pcol@prevdepth\maxdimen \pcol@setcurrcol}%
1580   {\@tempswatrue}%
1581   \pcol@measureupdate\@tempdima\dimen@ii\@tempdimc\pcol@prevdepth

```

Next we let $size_c(f) = 0$ if $\kappa_c(\tau^b)$ is void, or otherwise let $size_c(f) = h(\kappa_c(\tau^b)) + d(\kappa_c(\tau^b)) + \kappa_c(\tau^s)$ ¹⁹⁸ and `\if@tempswa = true` because $f_c(f) = true$ and thus $F_c(\{t, m, f\}) = true$. After that, we calculate $size_c(f) + size_c(b)$ by `\pcol@addflhd` giving it $\kappa_c(\lambda_b)$ and ∞ to mean `\textfloatsep` should be used for the calculation as its argument also to have `\if@tempswa = F_c(\{t, m, f, b\})`. Then we let $V_B = size_c(f) + size_c(b)$ if $V_B < size_c(f) + size_c(b)$, and let $V_P = size_c(t) + size_c(m) + size_c(f) + size_c(b) = SIZE_c(\{t, m, f, b\})$ and $c_{\max} = c$ if $F_c(\{t, m, f, b\}) = true$ and $V_P < SIZE_c(\{t, m, f, b\})$.

```

1582   \ifvoid\pcol@currfoot \dimen@z@
1583   \else
1584     \dimen@\ht\pcol@currfoot \advance\dimen@\dp\pcol@currfoot
1585     \advance\dimen@\skip\pcol@currfoot
1586     \@tempswatrue
1587   \fi
1588   \pcol@addflhd\@botlist\maxdimen
1589   \ifdim\dimen@>\@tempdimb \@tempdimb\dimen@ \fi
1590   \advance\dimen@\dimen@ii
1591   \if@tempswa \ifdim\dimen@>\@pageht
1592     \@pageht\dimen@ \@tempcntb\pcol@currcol
1593   \fi\fi

```

¹⁹⁸We ignore the height and depth of `\footnoterule` because they are expected to be 0 and are so in the default setting.

Next, we let d_c be the depth of the lowest non-empty items among the main vertical list, footnotes and bottom floats. That is, we let $d_c \leftarrow \kappa_c(\delta)$ at first, and then, if `\ifpcol@bfbottom = true`, override it by $d_c \leftarrow d(\kappa_c(\tau^b))$ if there are footnotes, and finally override it by $d_c \leftarrow 0$ for the bottom floats if exist adding `\textfloatsep` to $SIZE_c(\{t, m, f, b\})$ to have $SIZE_c(\{t, m, f, b'\})$. This overriding order of $d(\kappa_c(\tau^b))$ and then 0 by bottom floats is reversed when `\ifpcol@bfbottom = false` according to the implementation of `\@makecol`. Then, we invoke `\pcol@measureupdate` again to let $V'_P = SIZE_c(\{t, m, f, b'\})$ and $D_P = d_c$ if $F_c(\{t, m, f, b\}) = true$ and $V'_P < SIZE_c(\{t, m, f, b'\})$. It also lets $D_P = d_c$ if $F_c(\{t, m, f, b\}) = true$, $V'_P = SIZE_c(\{t, m, f, b'\})$ and $D_P > d_c$.

Finally, we let `\ifpcol@dfloats = true` if $\kappa_c(\lambda_d) \neq \emptyset$ to tell `\pcol@makeflushedpage` that the last page must be *full size* and `\pcol@output@end` to flush the deferred column-wise floats.

```

1594 \dimen@ii\pcol@prevdepth
1595 \ifpcol@bfbottom
1596 \ifvoid\pcol@currfoot\else \dimen@ii\dp\pcol@currfoot \fi
1597 \ifx\@botlist\@empty\else \dimen@ii\z@ \advance\dimen@\textfloatsep \fi
1598 \else
1599 \ifx\@botlist\@empty\else \dimen@ii\z@ \advance\dimen@\textfloatsep \fi
1600 \ifvoid\pcol@currfoot\else \dimen@ii\dp\pcol@currfoot \fi
1601 \fi
1602 \pcol@measureupdate\pcol@colht\dimen@\@pagedp\dimen@ii
1603 \ifx\@deferlist\@empty\else \pcol@dfloatstrue \fi
1604 \advance\pcol@currcol\@ne}

```

`\pcol@addflhd` The macro `\pcol@addflhd` $\langle list \rangle \langle tfs \rangle$ is invoked twice from `\pcol@measurecolumn` for a column c to measure $size_c(x)$ ($x \in \{t, b\}$) of top ($x = t$) or bottom ($x = b$) floats. The arguments and registers referred to in the macro have the following values according to $x = t$ or $x = b$.

	$x = t$	$x = b$
$\langle list \rangle$	$\kappa_c(\lambda_t)$	$\kappa_c(\lambda_b)$
$\langle tfs \rangle$	<code>\pcol@textfloatsep</code>	<code>\maxdimen</code>
<code>\if@tempswa</code>	<i>false</i>	$F_c(\{t, m, f\})$
<code>\dimen@</code>	0	$size(f)$

The macro is also used in `\pcol@makecol` and `\pcol@output@switch` for $x = t$ but with `\dimen@` having the height of `\pcol@prespan` for the measurement of the total height of pre-spanning-text stuff including top floats¹⁹⁹.

The macro at first examines if $\kappa_c(\lambda_x) = \langle list \rangle$ is empty and does nothing if so. Otherwise, `\if@tempswa` is turned *true* to have $f_c(t) = true$ for $x = t$ or $F_c(\{t, m, f, b\}) = true$ for $x = b$. Then we scan all floats in $\langle list \rangle$ applying `\pcol@hdflelt` to each float ϕ to add $h(\phi) + d(\phi) + \textfloatsep$ to `\dimen@`, from/to which we then subtract `\floatsep` and add σ_x because the last/first float is followed/preceded by the vertical skip of σ_x instead of `\floatsep`, to have $size_c(t)$ for $x = t$ or $size_c(f) + size_c(b)$ for $x = b$ being *returned* to `\pcol@measurecolumn`.

Note that σ_t is $\langle tfs \rangle = \textfloatsep$ if it is less than ∞ or `\textfloatsep` otherwise, while $\sigma_b = \textfloatsep$ always because $\langle tfs \rangle = \maxdimen$. Also note that σ_t can be biased by 10000pt and thus larger than 5000pt, if we have a MVL-float in top floats as discussed later. Another caution is that we ignore the contribution by `\topfigrule` nor `\botfigrule` because they should insert vertical items whose total height and depth are 0.

```

1605 \def\pcol@addflhd#1#2{%
1606 \ifx#1\@empty\else

```

¹⁹⁹In these invocations, `\if@tempswa` is meaningless and not examined by the invokers.

```

1607 \@tempswatruе
1608 \let\@elt\pcol@hdflelt
1609 #1\advance\dimen@-\floatsep
1610 \ifdim#2=\maxdimen \advance\dimen@\textfloatsep
1611 \else
1612 \advance\dimen@\pcol@textfloatsep
1613 \ifdim\pcol@textfloatsep>5000\p@ \advance\dimen@-\@M\p@ \fi
1614 \fi
1615 \let\@elt\relax
1616 \fi}
1617 \def\pcol@hdflelt#1{\advance\dimen@\ht#1\advance\dimen@\dp#1%
1618 \advance\dimen@\floatsep}

```

`\pcol@measureupdate` The macro `\pcol@measureupdate⟨V⟩⟨v⟩⟨D⟩⟨d⟩` is invoked twice in `\pcol@measurecolumn` for c to update $V \in \{V_T, V'_P\}$ and $D = \{D_T, D_P\}$ as follows if `\if@tempswa`, being $F_c(\{t, m\})$ for $V = V_T$ or $F_c(\{t, m, f, b\})$ for $V = V'_P$, is *true*.

$$V \leftarrow \max(V, v) \quad D \leftarrow \begin{cases} \min(D, d) & V = v \\ D & V \neq v \end{cases}$$

The arguments v and d have the followings according to V .

$$\begin{aligned} V = V_T : v &= SIZE_c(\{t, m\}) & d &= \delta_c \\ V = V'_P : v &= SIZE_c(\{t, m, f, b'\}) & d &= d_c \end{aligned}$$

```

1619 \def\pcol@measureupdate#1#2#3#4{\if@tempswa
1620 \ifdim#1<#2\relax#1#2\relax#3#4\relax
1621 \else\ifdim#1=#2\ifdim#3>#4\relax#3#4\fi\fi\fi\fi}
1622

```

`\pcol@synccolumn` The macro `\pcol@synccolumn` is invoked for each column $c \in [0, C)$ from `\pcol@sync` to set a synchronization point at V_T from the top of the current column-page of c if `\ifpcol@clear = false`, or flush it otherwise. After obtaining c 's column-context κ_c by `\pcol@getcurrcol`, we process one of the following three cases.

The first case is for flushing with `\ifpcol@clear = true`. In this case we simply add `\vfil` at the tail of the main vertical list in $\kappa_c(\beta^b)$ to make the whole column-page possibly with other items fit in a box of `\@colht` tall and, if $\kappa_c(\xi) \neq \infty$ to mean the column to be flushed has a synchronization point, we also add an infinite shrink of `1/10000fil` so as to cancel a finite shrink just below the point, as done in `\pcol@makecol`²⁰⁰. We also let $\kappa_c(\delta) = 1000\text{pt}$ to mimic \TeX 's mechanism of `\prevdepth` with the empty main vertical list in the next column-page²⁰¹.

```

1623 \def\pcol@synccolumn{%
1624 \pcol@getcurrcol
1625 \ifpcol@clear
1626 \global\pcol@prevdepth\@M\p@
1627 \global\setbox\@currbox\vbox{\unvbox\@currbox
1628 \ifdim\pcol@textfloatsep=\maxdimen \vfil
1629 \else \vskip\z@\@plus1fil\@minus.0001fil
1630 \fi}%

```

²⁰⁰Just in case, because it looks impossible that the natural height of the column-page exceeds $\pi^h(p_t)$ with pre-flushing column height check.

²⁰¹The author is not sure if this setting is really necessary but, at least, it looks working well (though other setting looks all right too).

The second and third cases are for synchronized column-switching. The second case is for $D_T = \infty$ to mean the synchronization point is set just below the top floats of a column whose main vertical list is empty because it is definitely $V_T \geq 0 > -\infty$. In this case, we should not put anything back to the main vertical list, because the column having defined the point will restart from the top of its column-page with `\topskip` and thus other columns should do so for the stuff following the point. Therefore, we put $\kappa_c(\beta^b)$ as the last top float, namely *MVL-float* because it is for the main vertical list, acquiring an `\insert` from `\@freelist` by `\@next` and assigning it to `\pcol@float` so that we pretend main vertical lists of all columns are empty.

The float has zero height and depth, and contains the followings if we have some real floats; a vertical skip of `-\floatsep` to go back to the bottom of the last real float; `\topfigrule` and a skip of `\textfloatsep` to separate $\kappa_c(\beta^b)$ from the last real float; and $\kappa_c(\beta^b)$ followed by `\vss` to avoid overfull and underfull. Otherwise, i.e., we don't have any real floats, neither of the skips nor `\topfigrule` are put in the MVL-float because we let `\floatsep = \textfloatsep = 0` and `\topfigrule = \relax` temporarily in a group. Then we set the synchronization point by enlarging the space below the MVL-float so that the total size of all floats including `\floatsep` and `\textfloatsep`, which may be 0 as set in the process above, is equal to V_T . This enlarging is done by letting $\kappa_c(\xi) = V_T - (v_c(t) - \textfloatsep + \floatsep)$, the second term of which is the vertical size of the top float sequence up to the MVL-float, and by replacing `\textfloatsep` with $\kappa_c(\xi)$ in the top float insertion process in `\pcol@cflt` so that the top floats including the MVL-float consumes V_T as a whole²⁰².

Note that the process above involves `\floatsep` and `\textfloatsep` with some finite stretch and shrink, but these factors will not contribute the final result because they are canceled by `\vss` in the MVL-float and by the small infinite stretch and shrink put by `\pcol@makecol` and this macro for flushing. Also note that $\kappa_c(\xi)$ is then biased by 10000pt so that `\pcol@cflt` will not put `\topfigrule` because it has been already put as a part of the MVL-float or we don't have any real floats. We also let $\kappa_c(\delta) = 1000$ to mean the column-page's main vertical list is empty, so as to mimic T_EX's mechanism of `\prevdepth` with an empty list again.

Another attention we have to pay is that column-pages with $\kappa_c(\xi) = \infty$ does not have any synchronization points, and thus $\kappa_c(\xi) < \infty$ means a synchronization has already taken place in them. If this $\kappa_c(\xi) < \infty$ happens with $D_T = \infty$ ²⁰³, we cannot update $\kappa_c(\xi)$ because `\pcol@measurecolumn` took care of its value on measuring $v_c(t)$. Therefore, we do nothing if $\kappa_c(\xi) < \infty$ but just let succeeding stuff be added to the main vertical list as in column-switching without synchronization.

```

1631 \else
1632   \@tempdimb\@tempdima
1633   \advance\@tempdimb-\skip\@currbox
1634   \ifdim\@tempdimc=\maxdimen
1635     \ifdim\pcol@textfloatsep=\maxdimen \begingroup
1636       \ifx\@toplist\@empty
1637         \textfloatsep\z@ \floatsep\z@ \let\topfigrule\relax
1638       \fi
1639       \pcol@Fb
1640       \@next\pcol@float\@freelist{\global\setbox\pcol@float\vbox to\z@{
1641         \vskip-\floatsep \topfigrule \vskip\textfloatsep
1642         \unvbox\@currbox \vss}}\pcol@ovf
1643       \pcol@Fe{synccolumn(topfloat)}%
1644       \@cons\@toplist\pcol@float

```

²⁰²This enlarging cannot be done by making the float's height $V_T - v_c(t) - \floatsep$ (or `\textfloatsep`) because the height can be negative.

²⁰³This can happen when a synchronization with $D_T = \infty$ is immediately followed by another synchronization or, more unlikely, by additions of items whose total amount is negative and then a synchronization.

```

1645     \advance\@tempdimb\textfloatsep \advance\@tempdimb-\floatsep
1646     \advance\@tempdimb\M\p@
1647     \global\pcol@prevdepth\@m\p@
1648     \global\pcol@textfloatsep\@tempdimb
1649     \endgroup \fi

```

The third and last case is for $D_T < \infty$ and thus most usual. In this case, we enclose everything in $\kappa_c(\beta^b)$ in a `\vbox` whose height is $h_c^v = V_T - v_c(t)$ and let $\kappa_c(\beta^b)$ have it so that the item following the synchronization point start at V_T . An attention we have to pay is that it can be $h_c^v < \text{\topskip}$ to let `\TeX` insert a vertical skip of the difference between them when the box is returned to the main vertical list pushing down the synchronization point a little bit²⁰⁴. Therefore, if so, we let $\kappa_c(\beta^b)$ have the followings; a `\vbox` of `\topskip` tall having its old contents at its top above which no vertical skip will be inserted; a vertical skip `-\topskip` going back to the page top; and a vertical skip h_c^v going down to the synchronization point.

The encapsulation of the old contents $\kappa_c(\beta^b)$ in the box of h_c^v tall gives us the following two features desirable for synchronization. First, all vertical glues in the box are *frozen*, nullifying finite stretches in them because we insert an infinite stretch of `1/10000fil` at the bottom of $\kappa_c(\beta^b)$ to push up its old contents respecting other infinite stretches if any, as done by `\raggedbottom`, and also nullifying finite and infinite shrinks because $h_c^v \geq v_c(m)$ definitely. This freezing and nullification keeps synchronization points already in $\kappa_c(\beta^b)$ from being observed moving a little bit vertically. That is, if we have a glue just below a synchronization point and it were *visible* to `\TeX`'s page builder, the item below the glue could move up/down when the builder found a break point with some shrink/stretch. Though this moving up/down is inhibited by the small infinite stretch/shrink which the column-page will at its bottom finally, it is undesirable to make `\TeX` misunderstanding that the glues are stretchable/shrinkable though they are not in reality.

Second, since the boxes in all column-page are zero deep due to the infinite stretch at their bottoms and these bottoms are aligned at the synchronization point, we have a clear view of the baseline progress after the synchronization regardless of their contents. That is, we let $\kappa_c(\delta) = D_T$ to *broadcast* D_T to all columns, so that the baselines of first items following the synchronization point are aligned `\baselineskip` below the bottom baseline of the column which defines D_T ²⁰⁵, if D_T plus the height of each item is less than `\baselineskip`.

In addition, we let $\kappa_c(\xi) = \text{\textfloatsep}$ to indicate the column-page has the synchronization point we just have set, if it was ∞ to mean the point is the first one. By this setting, `\pcol@makecol` and this macro itself will know that the column-page needs to have a small infinite shrink at its bottom to cancel finite ones below the synchronization point, while `\pcol@cflt` acts as `\ATEX's \cflt` because it should be $\kappa_c(\xi) \leq 5000\text{pt}$ to mean the column-page does not have a MVL-float.

```

1650     \else
1651     \global\pcol@prevdepth\@tempdimc
1652     \ifdim\pcol@textfloatsep=\maxdimen
1653     \global\pcol@textfloatsep\textfloatsep \fi
1654     \global\setbox\@currbox\vbox{%
1655     \ifdim\@tempdimb<\topskip

```

²⁰⁴This can happen not very unlikely especially with $v_c(t)$ a little bit less than V_T and $\kappa_c(\beta^b)$ being empty.

²⁰⁵Since D_T is given by one of the tallest columns whose depth is smallest among them, it is very likely that the bottom baseline of the column is lowest among all columns. However, another column can have the lowest one when its vertical size is a little bit shorter than V_T and its depth is small (e.g., 0). Though of course we can define D_T being V_T minus the height of the column having the largest height to make the first baseline below the synchronization point apart from the lowest one by `\baselineskip`, we dare to choose the definition of D_T because such lowest baseline often means that the column have some skip at its bottom to give us the impression that the space between the baselines of the tallest column and its first item is a little bit too large.

```

1656         \vbox to\topskip{\unvbox\@currbox \vskip\z@\@plus.0001fil}%
1657         \vskip-\topskip \vskip\@tempdimb
1658     \else
1659         \vbox to\@tempdimb{\unvbox\@currbox \vskip\z@\@plus.0001fil}%
1660     \fi}%
1661 \fi
1662 \fi

```

Finally, we let $\kappa_c(\nu_t) = 0$ to inhibit further addition of top floats because we have fixed the space for them²⁰⁶, and save it and other column-context members into κ_c by `\pcol@setcurrcol`.

```

1663 \global\@topnum\z@ \pcol@setcurrcol
1664 \advance\pcol@currcol\@ne}
1665

```

11.8 Page Flushing

`\pcol@output@flush` The macro `\pcol@output@flush` is invoked solely from `\pcol@specialoutput` to process the `\output` request made by `\flushpage`. We invoke `\pcol@makeflushedpage` giving it `\@colht` as the height of each column-page to have the ship-out image of the top page including its spanning stuff and page-wise footnotes in `\@outputbox` whose height is then set to be `\textheight`²⁰⁷, ensuring that its depth is capped by `\boxmaxdepth = \@maxdepth`. We also perform these height setting and depth capping on `\pcol@rightpage` if $C_L < C$ to mean parallel-paging. Then we invoke `\@outputpage` for shipping out, and then finally `\pcol@freshpage` to have a new page to start new column-pages in it.

```

1666 %% Special Output Routines: Page Flushing
1667
1668 \def\pcol@output@flush{%
1669   \pcol@makeflushedpage\@colht
1670   \pcol@Logstart\pcol@output@flush
1671   \setbox\@outputbox\vbox to\textheight{\boxmaxdepth\@maxdepth
1672     \unvbox\@outputbox}%
1673   \ifnum\pcol@ncolleft<\pcol@ncol
1674     \setbox\pcol@rightpage\vbox to\textheight{\boxmaxdepth\@maxdepth
1675       \unvbox\pcol@rightpage}%
1676   \fi
1677   \pcol@Logend\pcol@output@flush
1678   \@outputpage
1679   \pcol@freshpage}
1680

```

`\pcol@output@clear` The macro `\pcol@output@clear` is invoked solely from `\pcol@specialoutput` to process the `\output` request made by `\clearpage`. The first part up to `\@outputpage` and the last line of this macro are same as `\pcol@output@flush` to flush the top page and to have a newpage. In the remaining mid part, we invoke `\pcol@flushfloats` to ship out all deferred column-wise floats in all columns if any, and then do it for page-wise floats by the following invocations enclosed in a group; letting `\pcol@rightpage = \perp` for ordinary paging; `\@dblfloatplacement` to set up placement parameters followed by `\f@depth = 0` to nullify the setting `\f@depth = 1sp` possibly done by it as discussed in the item-(2) of §1.8; `\@makefcolumn` with `\@dbldeferlist` to have a float page in `\@outputbox` if any; and a loop of background painting of `\@outputbox`

²⁰⁶Allowing the addition is tremendously tough even when the column-page has sufficiently large space above the synchronization point.

²⁰⁷Just in case because the height of source `\@outputbox` should be exactly `\textheight` though not specified so on its construction in `\pcol@makeflushedpage`.

and, if $C_L < C$, of empty `\pcol@rightpage`, and `\@outputpage` followed by `\@makefcolumn` repeated while we have a float page, i.e., `\if@fcolmade = true`.

Note that the mid part is same as that found in `\@docclearpage` but we omit various adjuncts surrounding it as follows; examination of `\if@twocolumn` because we should have multiple columns; examination of `\if@firstcolumn` because we have to clear the page immediately even when we are not in the first column; concatenating `\@dbltoplist` with `\@dbldeferlist` and clearing it because the author believes `\@dbltoplist` must be empty on the invocation of this macro; and letting `\@colht = \textheight` because `\pcol@flushfloats` did it.

```

1681 \def\pcol@output@clear{%
1682   \pcol@makeflushedpage\@colht
1683   \pcol@Logstart\pcol@output@clear
1684   \setbox\@outputbox\vbox to\textheight{\boxmaxdepth\@maxdepth
1685     \unvbox\@outputbox}%
1686   \ifnum\pcol@ncolleft<\pcol@ncol
1687     \setbox\pcol@rightpage\vbox to\textheight{\boxmaxdepth\@maxdepth
1688       \unvbox\pcol@rightpage}%
1689   \fi
1690   \pcol@Logend\pcol@output@clear
1691   \@outputpage
1692   \pcol@flushfloats
1693   \begingroup
1694     \setbox\pcol@rightpage\box\voidb@x
1695     \@dblfloatplacement \let\f@depth\z@
1696     \@makefcolumn\@dbldeferlist
1697     \whilesw\if@fcolmade\fi{%
1698       \def\pcol@bg@floatheight{\pcol@bg@textheight}%
1699       \setbox\@outputbox\vbox to\textheight{%
1700         \pcol@bg@paintbox{Ff}\unvbox\@outputbox}%
1701       \ifnum\pcol@ncolleft<\pcol@ncol
1702         \setbox\pcol@rightpage\vbox to\textheight{\pcol@bg@paintbox{Ff}\vfil}%
1703       \fi
1704       \@outputpage
1705       \@makefcolumn\@dbldeferlist}%
1706   \endgroup
1707   \pcol@freshpage}
1708

```

`\pcol@makeflushedpage` The macro `\pcol@makeflushedpage<ht>` is invoked from `\pcol@output@flush` or `\pcol@output@clear` with $\langle ht \rangle = \@colht$ and from `\pcol@output@end` with $\langle ht \rangle = \pcol@colht$. At first, we invoke `\pcol@output@switch` with setting `\ifpcol@clear = true` to flush all pages up to $p_t - 1$ and to let $\kappa_c(\beta^b)$ have the ship-out image of the main vertical list of each column-page c in p_t . This invocation also lets `\pcol@colht = V'_P` so that hereafter we will refer V'_P through $\langle ht \rangle$ if it is `\pcol@colht` for last page. Then after obtaining p_t 's page context to have $page(p_t) = \pi^p(p_t)$, `\@colht = $\pi^h(p_t)$` and `\ifpcol@nospan` by `\pcol@getcurrpinfo`, we build the ship-out image of p_t in `\@outputbox`, and `\pcol@rightpage` if parallel-paging, taking special care of the last page case as follows.

- (1) If `\ifpcol@lastpage = false`, each of $\kappa_c(\beta^b)$ has ship-out image even if some or all of them are empty. It is unnecessary to be aware of the perfectly empty case because it should mean the page p_t is made blank intentionally.
- (2) If `\ifpcol@lastpage = true` but `\ifpcol@dfloats = true` too, the last page must have *full size* because we will have parallel columned pages having float columns for deferred

floats. However, if the page has nothing, i.e., $\pi^i(p_t) = \pi^f(p_t) = \perp$ and $V'_P = -\infty$, we must let `\@outputbox = \perp` (and `\pcol@rightpage = \perp` as well) to avoid an unnecessary blank page is shipped out. On the other hand, if $\pi^i(p_t) \neq \perp$ or $\pi^f(p_t) \neq \perp$ while $V'_P = -\infty$, we build a full size page as usual but letting `\@textbottom = \vfil` temporarily to avoid underfull in the process of building columns. Note that if $\pi^f(p_t) \neq \perp$, the page-wise footnotes are always put into `\@outputbox` regardless `\ifpcol@mgfnote` because the last page is not combined with post-environment stuff.

- (3) If `\ifpcol@lastpage = true`, `\ifpcol@dfloats = false` and $V'_P = -\infty$, we have to let `\@outputbox = \perp` unless $\pi^i(p_t) \neq \perp$ or $\pi^f(p_t) \neq \perp$ having non-merged footnotes. If $\pi^f(p_t)$ has non-merged footnotes, `\@outputbox` and `\pcol@rightpage` must have $\pi^f(p_t)$ possibly with $\pi^i(p_t)$ but without any columns, and must be put into the main vertical list as the leading part of post-environment stuff by modifying $V'_P = 0$. On the other hand $\pi^f(p_t) = \perp$ or it has merged footnotes, `\@outputbox` must have only $\pi^i(p_t)$ ²⁰⁸. Since page-wise floats become ordinary floats in post-environment stuff, we cannot paint its background and must remove `\dbltextfloatsep` at the bottom of $\pi^i(p_t)$.
- (4) If `\ifpcol@lastpage = true`, `\ifpcol@dfloats = false` and $V'_P > -\infty$, `\@outputpage` and `\pcol@rightpage` must have short columns of V'_P tall, together with $\pi^i(p_t)$ as in non-last pages but without $\pi^f(p_t)$ if it has merged footnotes.

To implement a part of special cares above, we at first let `\if@tempswa = true` iff `\ifpcol@lastpage = false`, $V'_P > -\infty$ or $\pi^f(p_t) \neq \perp$.

```

1709 \def\pcol@makeflushedpage#1{%
1710   \pcol@cleartrue \pcol@output@switch \pcol@clearfalse
1711   \pcol@getcurrpinfo{\global\c@page}{\global\@colht}\@tempskipa
1712   \ifpcol@lastpage \@tempswafalse \else \@tempswatruetrue \fi
1713   \ifdim\pcol@colht=-\maxdimen\else \@tempswatruetrue \fi
1714   \ifvoid\pcol@footins\else \@tempswatruetrue \fi

```

Next, if `\ifpcol@nospan = true` to mean the page p_t does not have spanning stuff in $\pi^i(p_t)$, we initialize both `\@outputbox` and `\pcol@rightpage` to be \perp . Otherwise, after letting `\if@tempswa = true` if `\ifpcol@dfloats = true` to make the last page full size if we are working on it as discussed in (2), we put $\pi^i(p_t)$ in `\@outputbox`, and paint its background by `\pcol@bg@paintbox` \edefining the height parameter `\pcol@bg@floatheight` with h being the height-plus-depth of $\pi^i(p_t)$ with the following two exceptions; one is the case of `\ifpcol@firstpage = true` to mean we are in starting page and thus the spanning stuff is pre-environment stuff having already been painted by `\pcol@output@start`; and the other is the case of `\if@tempswa = false` to mean we are working on a truly last page being empty except for the spanning stuff itself and thus the page-wise floats become a part of deferred floats in post-environment stuff as discussed in (3). In the latter exceptional case, excluding the case that the last page is also the starting page²⁰⁹, we also remove the last skip being `\dbltextfloatsep` so that those floats are naturally connected with other floats given in post-environment stuff also as discussed in (3). Then we *pack* the `\@outputbox` in itself by `\vbox` so that any stretch/shrink factors in it cannot affect the ship-out image especially when we paint its background²¹⁰. Then we do the similar procedure for `\pcol@rightpage` and make its height and depth equal to those of `\@outputbox`²¹¹. Finally we temporarily add h to `\topmargin` as done in `\pcol@ioutputelt`

²⁰⁸`\pcol@rightpage` must have the counterpart in right parallel-page if the spanning stuff is pre-environment stuff, while it is made \perp by `\pcol@output@end` if the spanning stuff are page-wise floats.

²⁰⁹Extremely exceptional because the closing environment does not have anything.

²¹⁰Though that hardly happens.

²¹¹If page-wise floats become a part of post-environment stuff's floats, `\pcol@rightpage` will be made \perp by `\pcol@output@end` afterward.

so that background painting of columns and so on with infinite extension can reach the paper top edge.

```

1715 \begingroup
1716 \ifpcol@nospan
1717 \global\setbox\@outputbox\box\voidb@x
1718 \global\setbox\pcol@rightpage\box\voidb@x
1719 \else
1720 \ifpcol@dfloats \@tempwatruet \fi
1721 \let\@elt\relax
1722 \edef\pcol@bg@floatheight{%
1723 \elt{\number\ht\pcol@spanning sp}\elt{\number\dp\pcol@spanning sp}}%
1724 \def\reserved@a{%
1725 \ifpcol@firstpage\else \if@tempswa \pcol@bg@paintbox{Ff}\fi\fi}%
1726 \@tempdima\ht\pcol@spanning \advance\@tempdima\dp\pcol@spanning
1727 \global\setbox\@outputbox\vbox{%
1728 \reserved@a \unvbox\pcol@spanning
1729 \ifpcol@firstpage\else \if@tempswa\else \unskip \fi\fi}%
1730 \global\setbox\@outputbox\vbox{\box\@outputbox}%
1731 \pcol@Fb
1732 \@cons\@freelist\pcol@spanning
1733 \pcol@Fe{makeflushedpage(spanning)}%
1734 \ifnum\pcol@ncolleft<\pcol@ncol
1735 \global\setbox\pcol@rightpage\vbox{%
1736 \ifpcol@paired\else \advance\c@page\@ne \fi
1737 \reserved@a \unvbox\pcol@rightpage}%
1738 \global\ht\pcol@rightpage\ht\@outputbox
1739 \global\dp\pcol@rightpage\dp\@outputbox
1740 \global\setbox\pcol@rightpage\vbox{\box\pcol@rightpage}%
1741 \fi
1742 \advance\topmargin\@tempdima
1743 \fi

```

Next, after `\globally` letting `\ifpcol@firstpage = false` because we will ship a page which may be the starting page shortly, we build the ship-out image of columns if required fundamentally by `\if@tempswa = true`. First, if the page p_t has page-wise footnotes, we shrink `\@colht = $\pi^h(p_t)$` by `\pcol@shrinkcolbyfn` to keep the room for the footnotes, to have $H = \@pageht$ being the possibly shrunk $\pi^h(p_t)$ for the reference in `\pcol@imakeflushedpage` after the further possible modification of `\@colht` we will make shortly. Second, if `\ifpcol@lastpage = true` but `\ifpcol@dfloats = true` too, we turn `\ifpcol@lastpage = false` because we need a full-sized last page, temporarily letting `\@textbottom = \vfil` if $V'_P = -\infty$ to avoid underfull due to perfectly empty column-pages as discussed in (2)²¹². Third, if we are working on a truly last page and $\langle ht \rangle < H$ to mean the tallest column is shorter than H , we let `\@colht = $\langle ht \rangle$` to let `\@makecol` build short column-pages. Note that it is definitely $\langle ht \rangle \leq H$ because the pre-flushing column height check on the last page makes that sure. Fourth and finally, unless all columns in truly last page are empty as discussed in (3), we invoke `\pcol@imakeflushedpage` $\langle C^0 \rangle \langle C^1 \rangle (b)$ once or twice, to put columns in right parallel-page to $b = \text{pcol@rightpage}$ with $[C^0, C^1] = [C_L, C]$ if $C_L < C$, and then to put left ones in $b = \text{@outputbox}$ with $[C^0, C^1] = [0, C_L]$ always²¹³.

```

1744 \global\pcol@firstpagefalse

```

²¹²Each column-page $cc_c(\beta^b)$ itself exists because the empty column-page has been visited by column-scan prior to `\output` request for environment closing.

²¹³The order of right to left is not essential in this macro but we follow the convention in `\pcol@outputelt`.

```

1745 \if@tempswa
1746 \ifvoid\pcol@footins\else
1747 \pcol@shrinkcolbyfn\@colht\pcol@footins\relax
1748 \fi
1749 \let\pcol@@hfil\relax \@pageht\@colht
1750 \ifpcol@lastpage \ifpcol@dfloats
1751 \ifdim\pcol@colht<\z@ \def\@textbottom{\vfil}\fi
1752 \pcol@lastpagefalse
1753 \fi\fi
1754 \ifpcol@lastpage \ifdim#1<\@colht \@colht#1\fi\fi
1755 \ifdim\@colht<\z@ \else
1756 \ifnum\pcol@ncolleft<\pcol@ncol
1757 \pcol@imakeflushedpage\pcol@ncolleft\pcol@ncol\pcol@rightpage
1758 \fi
1759 \pcol@imakeflushedpage\z@\pcol@ncolleft\@outputbox
1760 \fi
1761 \fi

```

After putting all column-pages, we examine if the page p_t has page-wise footnotes in $\pi^f(p_t)$. If so, and unless p_t is a truly last page and merged footnote typesetting is in effect to mean the page-wise footnotes will be merged with post-environment stuff, we put the footnotes in $\pi^f(p_t)$ below the column-pages. Prior to this however, we let `\pcol@fnheight@lpage` have the height-plus-depth of the footnote, so that `\pcol@output@end` know the size for the background painting of the footnotes, which `\pcol@imakeflushedpage` performed for non-last pages. We also put an empty box of the size into `\pcol@rightpage` by `\pcol@phantom` together with the `\skip` component of $\pi^f(p_t)$ to keep the space necessary especially when p_t is the last page. Then we put the footnotes in $\pi^f(p_t)$ into `\@outputpage` by `\pcol@putfootins`, reclaiming the contents of $\pi^f(p_t)$ and letting $\pi^f(p_t) = \perp$ so that `\pcol@output@end` will be unaware of the footnotes. We also let $V'_P = \text{\pcol@colht} = 0$ if p_t is a truly last page and it had $-\infty$ to indicate that the last page is not empty but has footnotes as discussed in (3).

```

1762 \gdef\pcol@fnheight@lpage{0pt}%
1763 \ifvoid\pcol@footins\else
1764 \@tempswatrue \ifpcol@lastpage \ifpcol@mgfnote \@tempswafalse \fi\fi
1765 \if@tempswa
1766 \pcol@Log\pcol@makeflushedpage{output}\pcol@footins
1767 \@tempdima\ht\pcol@footins \advance\@tempdima\dp\pcol@footins
1768 \xdef\pcol@fnheight@lpage{\number\@tempdima sp}%
1769 \ifnum\pcol@ncolleft<\pcol@ncol
1770 \global\setbox\pcol@rightpage\vbox{\unvbox\pcol@rightpage
1771 \vskip\skip\pcol@footins \nointerlineskip
1772 \pcol@phantom\pcol@footins \vskip\z@}%
1773 \fi
1774 \global\setbox\@outputbox\vbox{%
1775 \unvbox\@outputbox \pcol@putfootins\pcol@footins}%
1776 \pcol@Fb
1777 \@cons\@freelist\pcol@footins \gdef\pcol@footins{\voidb@x}%
1778 \pcol@Fe{makeflushedpage(pagefn)}%
1779 \ifdim\pcol@colht=-\maxdimen \global\pcol@colht\z@ \fi
1780 \fi
1781 \fi
1782 \endgroup}
1783

```

`\pcol@imakeflushedpage` The macro `\pcol@imakeflushedpage` $\langle C^0 \rangle \langle C^1 \rangle \langle b \rangle$ is invoked solely in `\pcol@makeflushedpage`

but can be twice with $(C^0, C^1, b) = (C_L, C, \text{\pcol@rightpage})$ if parallel-paging is in effect and with $(C^0, C^1, b) = (0, C_L, \text{\@outputbox})$ always, to build the ship-out image of the right or left parallel-page p_t in the box b already having spanning stuff or its blank counterpart if any, respectively.

After opening the `\vbox` of the ship-out image for b , at first we examine if `\ifpcol@paired = false` and $C^0 > 0$, and if so we temporarily increment $page(p_t)$ by one so that we check its parity for mirrored background painting correctly. Then if the page p_t has page-wise footnotes in $\pi^f(p_t)$, we paint its background, or that of its blank counterpart, by `\pcol@bg@paintbox` defining the parameter `\pcol@bg@footnoteheight` with the height-plus-depth of $\pi^f(p_t)$, as the very first element of the ship-out image as done in `\pcol@ioutputelt`, unless p_t is the truly last page for which the background painting is done in `\pcol@output@end`. Then we put spanning stuff in b itself if any.

Next, we invoke `\pcol@buildcolseprule` for column-separating rule drawing and background painting giving it H in `\@colht` possibly shrunk from $\pi^h(p_t)$ by page-wise footnotes, $[C^0, C^1)$ for the set of columns to be put, and `\@maxdepth` for non-last pages to paint the backgrounds of columns and column-separating gaps so that those of the last segment reach the page bottom, while for last page we give 0 to let the bottom be the real bottom of the columns. Then we put the painted backgrounds in `\@tempboxa` immediately.

```

1784 \def\pcol@imakeflushedpage#1#2#3{\global\setbox#3\vbox{%
1785   \ifpcol@paired\else\ifnum#1=\z@\else \advance\c@page\@ne \fi\fi
1786   \ifvoid\pcol@footins\else \ifpcol@lastpage\else
1787     \def\pcol@bg@footnoteheight{%
1788       \@elt{\ht\pcol@footins}\@elt{\dp\pcol@footins}}%
1789     \pcol@bg@paintbox{Nn}%
1790   \fi\fi
1791   \unvbox#3\nointerlineskip
1792   \ifpcol@lastpage \pcol@buildcolseprule\@colht#1#2\z@
1793   \else           \pcol@buildcolseprule\@colht#1#2\@maxdepth
1794   \fi
1795   \unvbox\@tempboxa

```

Now we put columns in a `\hbox` of $W_T = \text{\textwidth}$ wide. That is, for each c , being c' or $C-1-c'$ for the c' -th iteration determined by `\pcol@swapcolumn` according to the effectiveness of column-swapping and the parity of $page(p_t)$, we obtain c 's column-context κ_c by `\pcol@getcurrcol`, move $\kappa_c(\beta^b)$ into `\box255`, and let `\footins = \kappa_c(\tau)` by `\pcol@getcurrfoot` returning it to `\@freelist` if c has column-wise footnotes.

After that we examine if $\kappa_c(\rho_t) = \infty$ to mean we are working on the last page and the column-page is for a float column whose floats can be put as top floats, and let `\topfigrule = \relax` temporarily because the floats are not top ones in reality, if so. Note that the abnormal setting $\kappa_c(\rho_t) = \infty$ is not recovered because it will never be referred to and the register `\@toproom` it represents will be updated with correct value before it is referred to in post-environment stuff. Then we also check $\langle ht \rangle = H$ to mean the last page is full size. If both of them hold, the floats in $\kappa_c(\lambda_t)$ should be (or may be) put in the float column as usual and thus we put them in `\@outputbox` of H tall by `\pcol@makecolpage`. Otherwise we invoke `\pcol@makecol`²¹⁴, to have the ship-out image of the column-page in `\@outputbox`, possibly only for the deferred floats in $\kappa_c(\lambda_t)$ but without `\topfigrule` in this case. Note that we don't take care of the stretch/shrink of `\skip\footins` for page-wise footnotes because pre-flushing column height check on the column-page makes it sure that the natural height of the column-

²¹⁴Not `\pcol@makecol` because the main vertical list has `\vfil` and, if it has a synchronization point, a infinite shrink by `\pcol@synccolumn` at its tail already, and we should not do any special operations for page-wise footnotes. Also it is not `\@makecol` because we need to ensure the depth of resulting `\@outputbox` is capped.

page cannot be greater than \@colht . Also note that we give \@maxdepth to \pcol@makecol for non-last pages for depth capping, but for the last page we pass 0 to the macro because $H = \text{\@colht}$ should be large enough to accommodate everything in the column including its last box even if the box is unusually deep.

Then we put the \@outputbox above in a \hbox of \columnwidth wide preceded by \pcol@hfil being \relax for the first column, while it is $\text{\pcol@hfil}\langle c^g \rangle$, where $c^g = \text{\pcol@colsepid}$ being c or $c - 1$ without or with column-swapping respectively, to put a column-separating gap possibly with column-separating rule segments in \pcol@tempboxa built by $\text{\pcol@buildcolseprule}$. Finally, we save column-context especially those for float parameters into κ_c by $\text{\pcol@setcurrcolnf}$ because all column-wise footnotes have been shipped out.

```

1796 \hb@xt@\textwidth{%
1797   \@tempcntb#1\@whilenum\@tempcntb<#2\do{%
1798     \pcol@swapcolumn\@tempcntb\pcol@currcol#1#2\relax
1799     \pcol@getcurrcol
1800     \setbox\@cclv\box\@currbox
1801     \ifvoid\pcol@currfoot\else
1802       \pcol@Fb
1803       \@cons\@freelist\pcol@currfoot
1804       \pcol@Fe{imakeflushedpage(colfn)}%
1805     \fi
1806     \pcol@getcurrfoot\box
1807     \@tempswafalse
1808     \begingroup
1809     \ifdim\@toproom=\maxdimen
1810       \let\topfigrule\relax \ifdim\@colht=\@pageht \@tempswatrue \fi
1811     \fi
1812     \if@tempswa
1813       \pcol@Logstart{\pcol@makeflushedpage(1)}%
1814       \setbox\@outputbox\pcol@makecolpage
1815       \pcol@Logend{\pcol@makeflushedpage(1)}%
1816     \else
1817       \pcol@Logstart{\pcol@makeflushedpage(2)}%
1818       \ifpcol@lastpage \pcol@makecol\z@ \else \pcol@makecol\@maxdepth \fi
1819       \pcol@Logend{\pcol@makeflushedpage(2)}%
1820     \fi
1821     \pcol@hfil \hb@xt@\columnwidth{\box\@outputbox\hss}%
1822   \endgroup
1823   \edef\pcol@hfil{\noexpand\pcol@hfil{\pcol@colsepid}}%
1824   \pcol@setcurrcolnf
1825   \advance\@tempcntb\@ne}}}}
1826

```

\pcol@flushfloats The macro \pcol@flushfloats is invoked from $\text{\pcol@output@clear}$ and \pcol@output@
 $\text{\pcol@iflushfloats}$ end to flush all deferred column-wise floats in each column if any. After letting $\text{\@colht} = \text{\texttheight}$ for float columns, we iterate shipping out a page having float columns while $\text{\if@fcolmade} = \exists c \in [0, C) : (\kappa_c(\lambda_d) \neq ())$.

In the loop, we initialize $\text{\if@fcolmade} = \text{false}$, and then invoke $\text{\pcol@iflushfloats}$ twice or once according to $C_L < C$ or not to mean parallel-paging is in effect or not, respectively. That is, if $C_L < C$ we invoke the macro with $[C_L, C)$ and \pcol@rightpage for the right parallel-page, and do it with $[0, C_L)$ and \@outputbox always. Note that if $C_L = C$, we let $\text{\pcol@rightpage} = \perp$ to tell \@outputpage , which we invoke at the end of the loop to ship

out a page or a parallel-page pair, that the parallel-paging is not in effect.

The macro `\pcol@iflushfloats<C0><C1>` opens a `\vbox` to be set into b . Then if `\ifpcol@paired = false` and $C^0 > 0$ to mean we are working on a right non-paired parallel-page, we temporarily add `\c@page` by one for page parity examination for mirrored background painting. Then, the macro `\pcol@buildcolseprule` is invoked with `\@colht = \textheight` and $[C^0, C^1)$ for column-separating rule drawing in `\pcol@tempboxa` and background painting for columns and column-separating gaps in `\@tempboxa` put into b immediately.

Then we open a `\hbox` of `\textwidth` wide and initialize $f = \text{\if@tempswa}$ to be `\if@fcolmade`. Then for each $c \in [C^0, C^1)$, being c' or $C - 1 - c'$ for the c' -th iteration determined by `\pcol@swapcolumn` according to the effectiveness of column-swapping and the parity of `\c@page`, we put an inner `\hbox` of `\columnwidth = wc` wide preceded by `\pcol@hfil` being `\relax` at initial or `\pcol@hfil<cg>` otherwise for a column-separating gap and column-separating rule, where $c_g \in \{c, c-1\}$ without or with column-swapping respectively. That is, at first we obtain c 's column-context including $\kappa_c(\lambda_d)$ by `\pcol@getcurrcol` and pass $\kappa_c(\lambda_d)$ to `\@makefcolumn` to produce a float column in `\@outputbox` to be put into the inner `\hbox`. Then we do $f \leftarrow f \vee (\kappa_d(\lambda_d) \neq \emptyset)$ with $\kappa_d(\lambda_d)$ shrunk by `\@makefcolumn` to let f have $\exists c \in [0, C^1) : (\kappa_c(\lambda_d) \neq \emptyset)$ at the end of the loop for c , and then save the column-context into κ_c by `\pcol@setcurrcolnf` because we have no footnotes in c .

After the end of the loop, we move f to `\if@fcolmade` for the termination check of the loop in `\pcol@flushfloats`.

```

1827 \def\pcol@flushfloats{%
1828   \global\@colht\textheight
1829   \@whiles\if@fcolmade\fi{%
1830     \global\@fcolmadefalse
1831     \ifnum\pcol@ncolleft<\pcol@ncol
1832       \pcol@iflushfloats\pcol@ncolleft\pcol@ncol\pcol@rightpage
1833     \else
1834       \setbox\pcol@rightpage\box\voidb@x
1835     \fi
1836     \pcol@iflushfloats\z@\pcol@ncolleft\@outputbox
1837     \@outputpage}}
1838 \def\pcol@iflushfloats#1#2#3{\setbox#3\vbox{%
1839   \ifpcol@paired\else\ifnum#1=\z@\else \advance\c@page\@ne \fi\fi
1840   \pcol@buildcolseprule\@colht#1#2\@maxdepth \unvbox\@tempboxa
1841   \hb@xt@\textwidth{%
1842     \let\pcol@hfil\relax
1843     \if@fcolmade \@tempwatrue \else \@tempwafalse \fi
1844     \@tempcntb#1\@whilenum\@tempcntb<#2\do{%
1845       \pcol@swapcolumn\@tempcntb\pcol@currcol#1#2\relax
1846       \pcol@getcurrcol
1847       \@makefcolumn\@deferlist
1848       \pcol@hfil \hb@xt@\columnwidth{%
1849         \if@fcolmade \box\@outputbox \else \vbox to\@colht{\fi \hss}%
1850         \ifx\@deferlist\@empty\else \@tempwatrue \fi
1851         \edef\pcol@hfil{\noexpand\pcol@hfil{\pcol@colsepid}}%
1852         \pcol@setcurrcolnf
1853         \advance\@tempcntb\@ne}%
1854     \if@tempswa \global\@fcolmadetrue \else \global\@fcolmadefalse \fi}}}
```

`\pcol@freshpage` The macro `\pcol@freshpage` is invoked from `\pcol@output@flush` and `\pcol@output@clear` to start a new page after column flushing. At first, we let $p = p_b = p_t = 0$ and $\Pi = \emptyset$

because we know no pages are kept. Then we invoke `\pcol@startpage` to start a new page with a `\definition` of `\pcol@currpage = {}` to indicate the invoker is this macro (i.e., not `\pcol@opcol`). Then after keeping `\@colht` in $h = \pcol@colht$, we do the followings for each column $c \in [0, C)$.

First we obtain c 's column-context in κ_c by `\pcol@getcurrcol` but let $p = 0$ and `\@colroom = h`, which can be modified by $c' < c$, without referring to $\kappa_c(\beta^p)$ nor $\kappa_c(\beta^r)$ because they are obsolete. We also save `\@currbox` to `\pcol@currboxsave` because it may be modified by `\pcol@opcol` if we make float columns afterward. Then we invoke `\pcol@getcurrpage` to have the page context of $p = 0$, because it might be modified by a column $c' < c$ by producing float columns. After that and the invocation of `\pcol@floatplacement` for setting float parameters, we invoke `\pcol@startcolumn` for c 's column-page at $p = 0$, and iterate `\pcol@opcol` and `\pcol@startcolumn` while a float column is made by the latter²¹⁵. Note that we give the argument 0 to each invocation of `\pcol@startcolumn` to keep it from inserting deferred page-wise footnotes, which will be taken care of by `\pcol@restartcolumn` if any. At last in the loop, we restore `\@currbox` from `\pcol@currboxsave`, let $\kappa_c(\beta^b)$ be an empty `\vbox` because the main vertical list is empty, and save the column-context into κ_c by `\pcol@setcurrcolnf` because of no footnotes obviously, after saving p and `\@colroom`, which might be modified by the float column production, into $\kappa_c(\beta^p)$ and $\kappa_c(\beta^r)$.

After the loop above, finally we invoke `\pcol@restartcolumn` to return to the column in which `\flushpage` or `\clearpage` was issued.

```

1856 \def\pcol@freshpage{%
1857   \global\pcol@page\z@ \global\pcol@toppage\z@ \global\pcol@basepage\z@
1858   \global\let\pcol@pages\empty \global\let\pcol@currpage\empty
1859   \pcol@startpage \pcol@colht\@colht
1860   \pcol@currcol\z@ \@whilenum\pcol@currcol<\pcol@ncol\do{%
1861     \pcol@getcurrcol \pcol@page\z@ \@colroom\pcol@colht
1862     \let\pcol@currboxsave\@currbox
1863     \pcol@getcurrpage
1864     \pcol@floatplacement
1865     \pcol@startcolumn\z@
1866     \@whilesw@if@fcolmade\fi{\pcol@opcol \pcol@startcolumn\z@}%
1867     \let\@currbox\pcol@currboxsave
1868     \global\setbox\@currbox\vbox{}%
1869     \global\count\@currbox\pcol@page \global\dimen\@currbox\@colroom
1870     \pcol@setcurrcolnf
1871     \advance\pcol@currcol\@ne}%
1872   \pcol@restartcolumn}
1873

```

11.9 Last Page

`\pcol@output@end` The macro `\pcol@output@end` is invoked solely from `\pcol@specialoutput` to process the `\output` request made by `\endparacol`. We invoke `\pcol@makeflushedpage` for the last page production as the setting `\ifpcol@lastpage = true` done by `\endparacol` indicates, giving it $h = \pcol@colht$ in which `\pcol@sync` sets the height of the tallest column-page to have the ship-out image of the top page in `\@outputbox`.

Next, we define $\mathcal{M} = \pcol@mparbottom@out$ as follows. First, we invoke `\pcol@getmparbottom@last` giving it $y = V'_p - \ht\@outputbox$ being the negative counterpart of the height of spanning stuff in the last page, to let \mathcal{M} have the occupancy information of the

²¹⁵Each column can have deferred floats on the invocation from `\pcol@output@flush`.

bottom marginal note in each margin if any, or $mpar(y, y)$ otherwise. Then we transform y -coordinates in \mathcal{M} from those for columns to those for text area by `\pcol@bias@mpbout{-y}` to have the final result. Then to pass `\@mparbottom` for post-environment typesetting, we invoke `\pcol@do@mpbout` \defining `\pcol@do@mpbout@whole` to do nothing and `\pcol@do@mpbout@elem` to let `\@mparbottom = b` where $M_L^x = (mpar(t, b))$ and $x \in \{l, r\}$ according to the margin which post-environment marginal notes go to.

```

1874 %% Special Output Routines: Last Page
1875
1876 \def\pcol@output@end{%
1877   \pcol@Logstart\pcol@output@end
1878   \pcol@makeflushedpage\pcol@colht
1879   \@tempdima\pcol@colht \ifdim\pcol@colht<\z@ \@tempdima\z@ \fi
1880   \advance\@tempdima-\ht\@outputbox
1881   \pcol@getmparbottom@last\@tempdima
1882   \pcol@bias@mpbout{-\@tempdima}
1883   \def\pcol@do@mpbout@whole##1##2##3##4{\setbox\@tempboxa\hbox{##1##2##3##4}}%
1884   \def\pcol@do@mpbout@elem\@elt##1##2{\global\@mparbottom##2sp}%
1885   \pcol@do@mpbout

```

Next we process one of the following cases.

The first case is for `\ifpcol@dfloats = true` to mean the last page is followed by one or more pages having deferred column-wise floats and thus `\pcol@makeflushedpage` builds the ship-out image of the last page in *full size* in `\@outputbox` unless the page has nothing perfectly. Therefore, we ship the image out by `\@outputbox` unless it is \perp for perfectly empty case. Then we invoke `\pcol@flushfloats` to produce and ship out float pages, letting `\if@fcolmade = true` to tell the macro that at least we will have one float page. Now we have shipped out everything in the closing environment and thus we let `\ifpcol@output = false` to tell `\output` routine to work ordinarily. Then we let `\if@tempswa = fsp = true` to remember we will start a new page and thus `\@pagedp = $\delta = D_P = 1000$` to mimic TeX's `\prevdepth` mechanism. Finally we let `\@mparbottom = 0` and $\mathcal{M} = \mathcal{M}_0$ because no marginal notes are carried over to post-environment typesetting.

```

1886 \@tempswafalse
1887 \ifpcol@dfloats
1888   \ifvoid\@outputbox\else \@outputpage \fi
1889   \global\@fcolmadetrue \pcol@flushfloats
1890   \global\pcol@outputfalse
1891   \@tempswatrue \@pagedp\@m\p@ \global\@mparbottom\z@
1892   \global\let\pcol@mparbottom@out\pcol@mparbottom@zero

```

Before proceeding to the second and third cases, we let `\ifpcol@output = false` because we have nothing to ship out.

Then the second case is for $h = -\infty$ without deferred column-wise floats to mean all columns in the last page are empty and the page does not have non-merged page-wise footnotes. In this case, we examine if `\pcol@firstprevdepth = \relax` to mean we have had at least one new page in `paracol` environment, i.e., `\pcol@startpage` have been invoked at least once. If so, we let $f_{sp} = true$, $\delta = 1000$, `\@mparbottom = 0` and $\mathcal{M} = \mathcal{M}_0$ again and put nothing to the main vertical list so that the post-environment stuff starts from the top of the page. However, we have to take care of the case that $f_{ns} = false$ and thus `\@outputbox` has spanning stuff. If so, we acquire an `\insert` from `\@freelist` by `\@next` to let it have the spanning stuff, i.e., the contents of `\@outputbox`²¹⁶. Then the `\insert` is added to the head of `\@dbldeferlist`

²¹⁶It does not have `\dbltextfloatsep` at its tail because the skip has been removed by `\pcol@makeflushedpage`.

with the float placement code 10 to force L^AT_EX's float placement mechanism to put it to the page to be started shortly.

On the other hand, `\pcol@firstprevdepth ≠ \relax` means that it has `\prevdepth = δ'` just before `\begin{paracol}` in decimal integer representation. Since we have not started any pages in the environment, and all columns in the last page is empty, we have almost nothing in the environment. Note that the environment can have page-wise floats but they have not yet put into any pages but are kept in `\@dbldeferlist`, or merged footnotes but they are merged to those in post-environment stuff. Therefore, the pre-environment stuff and post-environment stuff must be *connected* naturally and thus we put the pre-environment stuff kept in `\@outputbox` to the main vertical list by `\unvbox`, letting $δ = δ'$ and keeping $f_{sp} = false$ ²¹⁷. In this case, the setting of `\mparbottom` and \mathcal{M} done at the beginning of this macro is correct because they describe the marginal notes in pre-environment stuff including `paracol` environments preceding it even if any.

```

1893 \else
1894 \global\pcol@outputfalse
1895 \ifdim\pcol@colht=-\maxdimen
1896 \ifx\pcol@firstprevdepth\relax
1897 \@tempwatrue \@pagedp\@m\p@ \global\@mparbottom\z@
1898 \global\let\pcol@mparbottom@out\pcol@mparbottom@zero
1899 \ifpcol@nospan\else
1900 \pcol@Fb
1901 \@next\@currbox\@freelist{\global\setbox\@currbox\box\@outputbox}%
1902 \pcol@ovf
1903 \pcol@Fe{output@end(spanning)}%
1904 \count\@currbox10\relax
1905 {\let\@elt\relax \xdef\@dbldeferlist{\@elt\@currbox\@dbldeferlist}}%
1906 \global\setbox\pcol@rightpage\box\voidb@x
1907 \fi
1908 \else \unvbox\@outputbox \@pagedp\pcol@firstprevdepth sp\relax
1909 \fi

```

The last case without deferred floats and with some non-empty columns or non-merged page-wise footnotes is most usual. In this case, we may simply put `\@outputbox` letting `\topskip = 0` because `\topskip` has already been inserted in column-pages or pre-environment stuff in the box²¹⁸.

However before putting the box back to the main vertical list, we have to take care of the background painting as follows. First we let `\ifpcol@havelastpage = true` to let `\@outputpage` paint the background of the post-environment stuff when the page having the last page completes. Second, we let `\pcol@bg@preposttop@left` have the height-plus-depth of the `\@outputbox` having the short last page because the background of post-environment stuff, or of pre-environment stuff if we have another `paracol` environment in the same page, to be painted is just below the box. We also `\pcol@bg@preposttop@right` have the same value but only if $C_L < C$, because otherwise we have to keep this macro unchanged so that the non-existent right parallel-page of the closing environment can be the post-environment stuff of a preceding environment and/or the pre-environment stuff of a succeeding one with parallel-paging. Note that in the aforementioned *fresh page* cases and the perfectly empty case, we may be unaware of these macros because it should have been made 0 by the last invocation of `\@outputpage` in the fresh page case or the pre-environment stuff and post-environment stuff are contiguous in the empty case.

²¹⁷The author of course know this situation is very unlikely but he is monomaniac.

²¹⁸If the last page has non-merged page-wise footnotes without any other items, `\topskip` has not been inserted, but this inconsistency without `\topskip` is acceptable.

Third and finally, we have to paint the background of non-merged page-wise footnotes because the painting is left by `\pcol@makeflushedpage` for this macro. Therefore, if `\pcol@fnheight@lpage > 0` to mean we have footnotes whose total height-plus-depth is in the macro, we paint their background by `\pcol@bg@paintbox` \defining `\pcol@bg@footnoteheight` with the size and temporarily re\defining `\pcol@bg@textheight` to be the height-plus-depth of `\@outputbox` because the footnotes are at the bottom of the box instead of the page. Note that the order of painting is right first and then left second if we have parallel-pages because we refer the height-plus-depth of `\@outputbox` being put into the main vertical list making the box \perp . Also note that if the right parallel-page is non-paired, we temporarily increment `\c@page` in `\pcol@rightpage` to let `\pcol@bg@paintbox` handle infinite extension to side margins correctly. Another remark is that we don't modify $\delta = \@pagedp$ and thus it keeps D_P in this case, and f_{sp} is kept *false*.

```

1910   \else
1911     \global\pcol@havelastpagetrue
1912     \@tempdima\ht\@outputbox \advance\@tempdima\dp\@outputbox
1913     \xdef\pcol@bg@preposttop@left{\number\@tempdima sp}%
1914     \ifnum\pcol@ncolleft<\pcol@ncol
1915       \global\let\pcol@bg@preposttop@right\pcol@bg@preposttop@left
1916     \fi
1917     \def\pcol@bg@textheight{\@elt{\ht\@outputbox}\@elt{\dp\@outputbox}}%
1918     \def\reserved@a{%
1919       \ifdim\pcol@fnheight@lpage>\z@
1920         \def\pcol@bg@footnoteheight{\@elt\pcol@fnheight@lpage}%
1921         \pcol@bg@paintbox{Nn}%
1922       \fi}%
1923     \ifnum\pcol@ncolleft<\pcol@ncol
1924       \global\setbox\pcol@rightpage\vbox{%
1925         \ifpcol@paired\else \advance\c@page\@ne \fi
1926         \reserved@a \unvbox\pcol@rightpage}%
1927     \fi
1928     \topskip\z@ \vbox{\reserved@a \unvbox\@outputbox}%
1929   \fi
1930 \fi

```

Now we have put almost everything in the last page but we may still have page-wise footnotes in $\pi^f(p_t)$ to be merged with those in post-environment stuff. Therefore, we `\insert` them through `\footins` as a part of post-environment stuff, and then do that for deferred footnotes in $\Phi = \pcol@topfnotes$ without using `\pcol@deferredfootins` because we don't need the height capping.

```

1931 \ifvoid\pcol@footins\else
1932   \pcol@Log\pcol@output@end{insert}\pcol@footins
1933   \pcol@Fb
1934   \insert\footins{\unvbox\pcol@footins}\@cons\@freelist\pcol@footins
1935   \pcol@Fe{output@end(pagefn)}%
1936 \fi
1937 \ifvoid\pcol@topfnotes\else \insert\footins{\unvbox\pcol@topfnotes}\fi

```

The following operations are for clean-up and set-up for the post-environment stuff; for all c , return $\kappa_c(\beta)$ obtained by `\pcol@getcurrcol` and $\gamma_0^c \neq \perp$ letting it \perp to `\@freelist`; reestablish the color stack by `\pcol@restorecolorstack` for column-0²¹⁹ so that the color stack is just Γ and is rewound at `\end{paracol}`, and let $\Gamma = \perp$; reload κ_d for $d = \pcol@lastcol$ being the column in which `\end{paracol}` occurs to let `\everypar = \kappa_d(\varepsilon)` and to let `\if@nbreak`

²¹⁹It can be any other column.

and `\if@afterindent` have the value represented by $\kappa_d(\sigma)$, so that the first paragraph of the post-environment stuff is typeset following them²²⁰; let `\pcol@prevdepth = δ` so that it is set to `\prevdepth` by `\pcol@invokeoutput`; let `\@colht = \@colroom = \textheight` to mean the single-column page does not have any floats so far because those produced in or before the environment have already been shipped out, are put to the main vertical list packed in `\@outputbox`, or are in `\@dbldeferlist`.

As for deferred page-wise floats produced in the environment, including those once put in the last page but returned to the list by the operation described above, we move them to `\@deferlist` because they are now column-wise floats. Then we invoke `\pcol@floatplacement` to reinitialize float placement parameters. Finally, if $f_{sp} = true$, we invoke `\@startcolumn` and then repeat invocations of `\@opcol` and `\@startcolumn` while float pages are produced, after letting `\ifpcol@lastpage = false` to make `\@combinefloats` work as L^AT_EX's original²²¹.

```

1938 \pcol@currcol\z@ \@whilenum\pcol@currcol<\pcol@ncol\do{%
1939   \pcol@Fb
1940   \pcol@getcurrcol \@cons\@freelist\@currbox
1941   \ifvoid\pcol@ccuse{@box}\else
1942     \@cons\@freelist{\pcol@ccuse{@box}}%
1943     \pcol@ccxdef{\voidb@x}%
1944   \fi
1945   \pcol@Fe{output@end(col)}%
1946   \advance\pcol@currcol\@ne}%
1947 \pcol@currcol\z@ \pcol@restorecolorstack
1948 \global\setbox\pcol@colorins\box\voidb@x
1949 \pcol@currcol\pcol@lastcol\relax \pcol@getcurrcol
1950 \global\pcol@prevdepth\@pagedp
1951 \global\@colht\textheight
1952 \global\@colroom\textheight
1953 \global\let\@deferlist\@dbldeferlist \gdef\@dbldeferlist{}%
1954 \pcol@floatplacement
1955 \pcol@lastpagefalse
1956 \if@tempswa
1957   \@startcolumn \@whilesw\if@fcolmade\fi{\@opcol\@startcolumn}%
1958 \fi
1959 \pcol@Logend\pcol@output@end
1960 }
1961

```

12 Starting Environment

`\pcol@invokeoutput` Before giving the definition of `paracol` environment and commands used in it, we define the macro `\pcol@invokeoutput<pen>` invoked from them to make an `\output`-request with `\penalty` of $\langle pen \rangle = \pcol@op@f$. The macro's structure is similar to that for the request sequence in `\end@float` as follows; insert a `\penalty` of -10004 to save the main vertical list in `\@holdpg`; save `\prevdepth`; insert a `\vbox` to make it sure the following `\penalty` of $\langle pen \rangle$ is kept even when we are at the top of a page; and finally restore `\prevdepth`.

A difference is that we zero-clear `\deadcycles` because it can be reach `\maxdeadcycles = 100` easily if a page has many synchronizations and many columns. Another and more important

²²⁰For rare cases that the last item of the closing environment is a sectioning command, but a user has such very unusual usage.

²²¹Letting `\pcol@textfloatsep = ∞` is done by `\pcol@floatplacement`.

difference is in the save/restore of `\prevdepth`. First, the value of the register is saved in our own `\pcol@prevdepth` instead of `\@tempdima` by a `\global` assignment so that `\output`-routine refers to it²²². Second, the value restored to `\prevdepth` may be different from that we just have saved because `\output`-routine may update `\pcol@prevdepth` to have, for example, the value saved in $\kappa_c(\delta)$ when we left from c which we are now restarting.

In addition to above, after the invocation of `\output`-routine, we let `\linewidth = w_c - \mu` so that the register is shrunk from w_c by the total width of left and right margins of the `list`-like environment surrounding `paracol` if $\mu = \text{\pcol@lrmargin} > 0$ to mean that. Then if so, we set `\parshape` to let every line of paragraphs in the column c is indented by `\@totalleftmargin` and has width of `\linewidth`, as L^AT_EX's `list` does. We also let `\hsize = w_c` because it should have the width of the column even in a `list`-like environment.

The macro is invoked from `\pcol@zparacol` ($f = \text{start}$), `\pcol@switchcol` for `\switchcolumn` and column-switching environments ($f = \text{switch}$), `\pcol@visittallcols` for column-scan prior to synchronized column-switching and page flushing ($f = \text{switch}$), `\pcol@flushclear` for pre-flushing column height check ($f = \text{switch}$), `\pcol@com@flushpage` for `\flushpage` ($f = \text{flush}$), `\pcol@com@clearpage` for `\clearpage` ($f = \text{clear}$), and `\endparacol` ($f = \text{end}$).

```

1962 %% Starting Environment
1963
1964 \def\pcol@invokeoutput#1{\deadcycles\z@
1965   \pcol@Logstart{\pcol@invokeoutput
1966     {#1:\the\pcol@currcol/\the\pcol@nextcol%
1967       \ifnum#1=-10011:\ifpcol@sync s\fi \ifpcol@clear c\fi\fi}}%
1968   \penalty-\@Miv \global\pcol@prevdepth\prevdepth \vbox{}}%
1969   \penalty#1\relax \prevdepth\pcol@prevdepth
1970   \linewidth\columnwidth \advance\linewidth-\pcol@lrmargin
1971   \ifdim\pcol@lrmargin>\z@ \parshape\@ne\@totalleftmargin\linewidth \fi
1972   \hsize\columnwidth
1973   \pcol@Logend{\pcol@invokeoutput{#1}}}}
1974

```

`\paracol` The API macro `\paracol[C_L][*]{ C }[text]` is invoked by `\begin{paracol}` to start a `\pcol@xparacol` `paracol` environment. The macro simply examines the existence of the optional argument `\pcol@yparacol` C_L , whose default value C is given by `\pcol@xparacol`, to decide the number of columns in `\pcol@zparacol` left parallel-pages. Then if the optional argument is given, `\pcol@yparacol` examines the existence of ‘*’ following it for non-paired parallel-paging to let `\ifpcol@paired = false`, while `\paracol` gave the default `true` for paired one to the switch. Then the all other operations to start the environment is done by `\pcol@zparacol`.

```

1975 \def\paracol{\global\pcol@pairedtrue \@ifnextchar[%]
1976   \pcol@yparacol\pcol@xparacol}
1977 \def\pcol@xparacol#1{\pcol@zparacol[#1]{#1}}
1978 \def\pcol@yparacol[#1]{%
1979   \@ifstar{\global\pcol@pairedfalse \pcol@zparacol[#1]}%
1980     {\pcol@zparacol[#1]}}

```

In `\pcol@zparacol`, after making it sure to be in vertical mode by `\par`, at first we examine if we are neither in a box by `\ifinner` nor with ordinary two-column typesetting by `\if@twocolumn`, and complain about inappropriateness unless our expectation is satisfied. Then we let C_L and C have the value given through the corresponding arguments, unless $C_L > C$ to let us make $C_L = C$ silently. Next we examine $C_L < C$, and if not we let

²²²The assignment is not necessary to be done `\globally` but we dare to do it to make all assignments to `\pcol@prevdepth` being `\global` consistently.

`\ifpcol@paired = true` regardless the setting in `\pcol@yparacol` because non-paired typesetting is meaningless without parallel-paging. On the other hand, if non-paired typesetting is specified, we let `\ifpcol@swapcolumn = false` but *not* `\globally` because column-swapping is meaningless in non-paired mode in the environment now starting.

Second, we perform the operations done by `\item` if `\if@newlist = true` and `\if@inlabel = false` to mean the first one in a list-like environment will appear at the very first line of the leftmost column. That is, we invoke `\@nbitem` if `\if@nobreak = true`, or add a penalty `\@beginparpenalty` and a vertical skip `\@topsep-\parskip-\itemsep` so that the first `\item` is `\@topsep` apart from the last line above the environment, and then let `\if@newlist = false`. The reason why we do these operations here is that, if the `paracol` environment is enclosed in a list-like environment without anything between two `\begin` for environments, we have to align all first `\items` in all columns. That is, if we did not do that, the *literally first* `\item` would do that resulting in the column having the `\item` led by the vertical skip of `\@topsep` while others should have ordinary inter-`\item` skips. Therefore, we perform the operations on behalf of all first `\items` in all columns to have the skip of `\@topsep` *above* the `paracol` environment we are opening. Note that if `\begin{paracol}` immediately follows a `\begin` for a `trivlist`-like environment, `\if@inlabel = true` because the first `\item` was given in the opening macro and thus the operations shown above has already been performed.

```

1981 \def\pcol@zparacol[#1]#2{\par
1982   \ifinner \@parmoderr \fi
1983   \iftwocolumn \PackageError{paracol}{%
1984     Environment paracol cannot work with ordinary two-column
1985     typesetting.}\@ehb\fi
1986   \global\pcol@ncolleft#1\relax \global\pcol@ncol#2\relax
1987   \ifnum\pcol@ncolleft>\pcol@ncol \global\pcol@ncolleft\pcol@ncol \fi
1988   \ifnum\pcol@ncolleft<\pcol@ncol\else \global\pcol@pairedtrue \fi
1989   \ifpcol@paired\else \pcol@swapcolumnfalse \fi
1990   \if@newlist \if@inlabel\else
1991     \if@nobreak \@nbitem
1992     \else
1993       \addpenalty\@beginparpenalty
1994       \addvspace\@topsep
1995       \addvspace{-\parskip}\addvspace{-\itemsep}%
1996     \fi
1997     \global\@newlistfalse
1998   \fi\fi

```

The third operation group is to set up lists for counters as follows. First we obtain $\Theta = \text{\c1@ckpt}$ to have all counters defined by `\newcounter` and `page`. Then we scan $\Theta^g = \text{\pcol@gcounters}$ applying `\pcol@remctrelt` to each element to have $\Theta^l = \text{\pcol@counters}$ for all local counters by removing all $\theta^g \in \Theta^g$ from Θ , and to move $\zeta(\theta^g) = \text{\c1@}\theta^g$ to `\pcol@c1@` θ^g redefining `\c1@` $\theta^g = \text{\pcol@stepcounter}\{\theta^g\}$ to clear descendant local counters of θ^g , i.e., those in $\zeta(\theta^g)$.

Next we scan Θ^l applying `\pcol@thectrelt` to each element $\theta^l \in \Theta^l$ to let `\pcol@thectr@` $\theta^l = \text{\the}\theta^l$ so that the former has the *default* local representation of θ^l . The macro `\pcol@thectrelt` also lets `\the` $\theta^l = \text{\pcol@thectr@}\theta^l\cdot 0$ if the local representation for θ^l and $c = 0$ has been defined by `\definethecounter`. This is necessary because in the first visit to the leftmost column 0 we will neither invoke `\pcol@switchcol` nor thus scan Θ^l with `\pcol@setctrelt` which defines local representations, unless a spanning text is specified with `\begin{paracol}`.

Next we give the initial value of $val_c(\theta^l)$ for each column c and local counter $\theta^l \in \Theta^l$ by the followings enclosed in a group. First we scan Θ_0 applying `\pcol@loadctrelt` to each

$\theta^l \in \Theta'_0 = \{\theta \mid \langle \theta, val_0(\theta) \rangle \in \Theta_0\}$ to have the value $val_0(\theta^l)$ in `\pcol@ctr@ θ^l` temporarily. Next we scan Θ^l to pick local counters θ such that $val(\theta) \neq val_0(\theta)$ where $val(\theta)$ is the value of θ outside `paracol` environment, or $\theta \notin \Theta'_0$, to let them be listed in `\@gtempa`. That is, we pick local counters which have been updated or defined after the closing of the previous `paracol`, or everything in Θ^l at the first `\begin{paracol}` because $\Theta_0 = \emptyset$ ²²³. Finally, we invoke `\pcol@synccounter` giving it `\@gtempa` as its argument to let $val_c(\theta) = val(\theta)$ for all c and for each θ in `\@gtempa`. That is, when `paracol` environment appears two or more times, the value of a local counter at the beginning of the second or succeeding environment is kept unchanged from that at the end of the previous environment, unless it has been updated between them. Note that these scans above are enclosed in a group in order to make `\pcol@ctr@ θ^l` local and thus collected as a garbage at `\endgroup`.

```

1999 \global\let\pcol@counters\cl@@ckpt
2000 \let\@elt\pcol@remctrelt \pcol@gcounters
2001 \let\@elt\pcol@thectrelt \pcol@counters
2002 \begingroup
2003 \let\@elt\pcol@loadctrelt \csname pcol@counters0\endcsname
2004 \let\@elt\pcol@cmpctrelt \global\let\@gtempa\empty \pcol@counters
2005 \pcol@synccounter\@gtempa
2006 \endgroup

```

Fourth, we set up a few L^AT_EX's typesetting parameters which should have appropriate values in the environment. We let `\if@twocolumn = true` so that `float*` environments work for page-wise floats and LaTeX's `\@addmarginpar` determine the margin for marginal notes by `\if@firstcolumn` whose truth value is determined by our own `\@addmarginpar`. We also let `\col@number = 1` instead of C so that its (almost surely) sole user `\maketitle` will not produce the title with `\twocolumn` which cannot be in the environment.

Then we invoke `\pcol@setcolumnwidth$C^0$$C^1$$r$$s$` once or twice with $(C^0, C^1, r, s) = (0, C_L, \text{\pcol@columnratioleft}, \text{\pcol@colwidthspecleft})$ always for left parallel-page, and with $(C_L, C, \text{\pcol@columnratioreight}, \text{\pcol@colwidthspecright})$ for right one if $C_L < C$ to define column widths $w_c = \text{\pcol@columnwidth} \cdot c$ and that of column-separating gaps $g_c = \text{\pcol@columnsep} \cdot c$ for all $c \in [0, C) = [0, C_L) \cup [C_L, C)$.

Then we initialize other variables as follows; $\mu = \text{\pcol@lrmargin} = \text{\textwidth} - \text{\linewidth}$ so that, if $\mu > 0$, `\linewidth` for c has $w_c - \mu$ reflecting the paragraph shape of the list-like environment surrounding `paracol` environment; `\pcol@topskip = \topskip` for the second and succeeding pages; `\pcol@textfloatsep = \infty` to mean we don't have any synchronization points so far; `\ifpcol@lastpage = false` because the starting page is not the last so far; and `\pcol@firstprevdepth = \prevdepth` in decimal integer form for the extreme empty case.

We also make the macro `\@combinefloats` `\let`-equal to our own `\pcol@combinefloats` throughout the environment, after saving its original definition into `\pcol@@combinefloats` for processing `\output` request sneaked from outside of environment, so that our customization is in effect for any invocations including those from L^AT_EX's own macros. Similarly, `\@addmarginpar` is made `\let`-equal to our own `\pcol@addmarginpar` while its original definition is saved into `\pcol@@addmarginpar` but in this case we need the original for the implementation of our own. On the other hand, `\end@dblfloat` is simply replaced with our own `\pcol@end@dblfloat` being what L^AT_EX had had until 2014 as discussed in item-(1) of §1.8.

```

2007 \global\@twocolumntrue \col@number\@ne
2008 \pcol@setcolumnwidth\z@\pcol@ncolleft
2009 \pcol@columnratioleft\pcol@colwidthspecleft

```

²²³Undefined in fact.

```

2010 \ifnum\pcol@ncolleft<\pcol@ncol
2011   \pcol@setcolumnwidth\pcol@ncolleft\pcol@ncol
2012     \pcol@columnratoright\pcol@colwidthspecright
2013 \fi
2014 \pcol@lrmargin\textwidth \advance\pcol@lrmargin-\linewidth
2015 \global\pcol@topskip\topskip
2016 \global\pcol@textfloatsep\maxdimen
2017 \pcol@lastpagefalse \xdef\pcol@firstprevdepth{\number\prevdepth}%
2018 \let\pcol@@combinefloats@combinefloats \let\@combinefloats\pcol@combinefloats
2019 \let\pcol@@addmarginpar\@addmarginpar \let\@addmarginpar\pcol@addmarginpar
2020 \let\end@dblfloat\pcol@end@dblfloat

```

Fifth, we save the original definition of `\set@color` into `\pcol@set@color`, and then examine if `\set@color` \neq `\relax` meaning some coloring package is loaded. If any coloring packages are not loaded, we make macros for background painting, namely `\pcol@bg@paintpage`, `\pcol@bg@paintcolumns` and `\pcol@bg@paintbox`, `\let`-equal to `\relax` for the first two and to `\@gobble` for the last so that they do nothing without coloring package.

If the coloring is enabled, on the other hand, we redefine L^AT_EX's `\set@color` so that it works as `\pcol@set@color@push` with color stack. We also prepare the text coloring mechanism to let `\ifpcol@inner = true` in every `\vbox` in the `paracol` environment as follows. First, we let `\ifpcol@inner = false` because we are not in any `\vboxes` obviously. Then, to use `\everyvbox` for our own purpose, we do the followings; (1) `\glo` globally assign a token `\pcol@dummysymbol`, which should never occurs, to `\pcol@everyvbox`; (2) save `\everyvbox` into `\pcol@everyvbox` locally; (3) let `\everyvbox` have a `\the`-reference to `\pcol@everyvbox` and `\pcol@innerture` to let `\ifpcol@inner = true`; and then (4) make `\everyvbox` `\let`-equal to `\pcol@everyvbox`. By the last operation (4), any `\everyvbox` appearing in the `paracol` environment is replaced with `\pcol@everyvbox` to keep the real `\everyvbox` from modified nullifying our own operation `\pcol@innertrue`. On the other hand, since both `\everyvbox` and `\pcol@everyvbox` are registers to hold tokens and thus any operations applicable to `\everyvbox` are also applicable to `\pcol@everyvbox`, any updates on `\everyvbox` and explicit references to it are simulated by `\pcol@everyvbox`. Then the initial tokens given to `\pcol@everyvbox` by the saving operation (2) or tokens given inside the environment are correctly processed whenever a `\vbox` is opened, together with `\pcol@innertrue` to fulfill our own purpose, because the real `\everyvbox` is let have a `\the`-reference to `\pcol@everyvbox` by (3). We also reserve the invocation of `\pcol@restoreeveryvbox` by `\aftergroup` so that the macro is invoked just after `\end{paracol}` to examine if any `\glo` global assignments to `\everyvbox` has been made in the environment. The funny `\glo` global assignment (1) is done for this examination so that we detect global assignments done in the environment having been closed because they should have changed the global value of `\pcol@everyvbox` to something different from `\pcol@dummysymbol`.

Then we continue the case having some coloring package to make background painting macros `\pcol@bg@paintpage`, `\pcol@bg@paintbox` and `\pcol@bg@paintcolumns` activated by making them `\let`-equal to their `@@` counterparts namely `\pcol@bg@@paintpage`, `\pcol@bg@@paintbox` and `\pcol@bg@@paintcolumns` which implement background painting.

Finally we empty the shadow color stack $\hat{T} = \pcol@colorstack@shadow$ to give it initial value regardless of the availability of coloring package.

```

2021 \global\let\pcol@set@color\set@color
2022 \ifx\set@color\relax
2023   \let\pcol@bg@paintpage\relax \let\pcol@bg@paintbox\@gobble
2024   \let\pcol@bg@paintcolumns\relax
2025 \else
2026   \let\set@color\pcol@set@color@push
2027   \pcol@innerfalse

```

```

2028 \global\pcol@everyvbox{\pcol@dummysymbol}%
2029 \pcol@everyvbox\everyvbox
2030 \everyvbox{\the\pcol@everyvbox \pcol@innertrue}
2031 \let\everyvbox\pcol@everyvbox
2032 \aftergroup\pcol@restoreeveryvbox
2033 \let\pcol@bg@paintpage\pcol@bg@@paintpage
2034 \let\pcol@bg@paintbox\pcol@bg@@paintbox
2035 \let\pcol@bg@paintcolumns\pcol@bg@@paintcolumns
2036 \fi
2037 \gdef\pcol@colorstack@shadow{ }%

```

The sixth settings are for (mainly column-wise) footnotes. We initialize two footnote-related count registers letting $b_f = \text{\pcol@footnotebase}$ have \c@footnote and zero-clearing $n_f = \text{\pcol@nfootnotes}$. Then we redefine L^AT_EX's API macros \footnote , \footnotemark and \footnotetext to let them be our own \pcol@footnote , $\text{\pcol@footnotemark}$ and $\text{\pcol@footnotetext}$ so that they have starred-versions. The other API macro to be redefined, if page-wise footnote typesetting is in effect, is \footnoterule which lets $\text{\columnwidth} = \text{\textwidth}$ before invoking its original version saved in $\text{\pcol@footnoterule}$ so that it acts as in single-columned typesetting rather than multi-columned. Then we redefine L^AT_EX's internal macro \@footnotetext ²²⁴ letting it be our own \pcol@fntext for encapsulating a footnote in a \vbox and for deferred \insertion with page-wise footnote typesetting.

```

2038 \pcol@footnotebase\c@footnote \global\pcol@nfootnotes\z@
2039 \let\footnote\pcol@footnote
2040 \let\footnotemark\pcol@footnotemark
2041 \let\footnotetext\pcol@footnotetext
2042 \ifpcol@scfnote
2043 \def\footnoterule{{\columnwidth\textwidth \pcol@footnoterule}}%
2044 \fi
2045 \ifundefined{H@@footnotetext}{%
2046 \let\@footnotetext\pcol@fntext
2047 }{%
2048 \let\H@@footnotetext\pcol@fntext
2049 }

```

Seventh, we let \marginpar , \@mn@@marginnote and \@xympar be our own versions \pcol@marginpar , \pcol@marginnote and \pcol@xympar respectively for the emulation of \marginnote , saving the original version of the first and third into \pcol@@marginpar and \pcol@@xympar . Then we inactivate API macros \twosided and $\text{\footnoteplacement}$ together with their backward-compatible macros $\text{\swapcolumnninevenpages}$, $\text{\noswapcolumnninevenpages}$, \footnotelayout , $\text{\multicolumnfootnotes}$, $\text{\singlecolumnfootnotes}$ and \mergedfootnotes , so that they commonly invoke \pcol@ignore because their inherent operations turning corresponding switches are harmful in \paracol environment. Note that the inactivation of \twosided is done by redefinition of \pcol@twosided because we need optional argument processing by \twosided even when it is inactivated. Further note, that since \footmisc defines its own version of \footnotelayout , we must only redefine \footnotelayout if it is what we expect, i.e. an alias to $\text{\footnoteplacement}$.

```

2050 \let\pcol@@marginpar\marginpar \let\marginpar\pcol@marginpar
2051 \let\@mn@@marginnote\pcol@marginnote
2052 \let\pcol@@xympar\@xympar \let\@xympar\pcol@xympar
2053 \def\pcol@twosided[#1]{\pcol@ignore\twosided}%
2054 \def\swapcolumnninevenpages{\pcol@ignore\swapcolumnninevenpages}%
2055 \def\noswapcolumnninevenpages{\pcol@ignore\noswapcolumnninevenpages}%

```

²²⁴This is called \H@@footnotetext if \hyperref (with option \hyperfootnote) is loaded.

```

2056 \def\footnoteplacement#1{\pcol@ignore\footnoteplacement}%
2057 \ifx\footnotelayout\footnoteplacement
2058   \def\footnotelayout#1{\pcol@ignore\footnotelayout}%
2059 \fi
2060 \def\multicolumnfootnotes{\pcol@ignore\multicolumnfootnotes}%
2061 \def\singlecolumnfootnotes{\pcol@ignore\singlecolumnfootnotes}%
2062 \def\mergedfootnotes{\pcol@ignore\mergedfootnotes}%

```

Eighth, we scan the list `\pcol@localcommands` of $\langle com \rangle$ being the name of commands, e.g., `switchcolumn`, available only in the environment or customized for the environment, applying `\pcol@defcomelt` to each $\langle com \rangle$ to let $\backslash\langle com \rangle = \backslashpcol@com@.\langle com \rangle$ the latter of which is the real implementation of the former. Note that the list does not have all environment-local API commands but we omit `\column(*)` for `column(*)` environments because their implementations `\pcol@com@column(*)` are modified after the first invocation. Therefore, we `\define` `\column(*)` to have `\pcol@com@column(*)` in their bodies²²⁵. We also give the first `\definitions` of `\pcol@com@column(*)` to let them do nothing but re`\define` themselves by `\pcol@defcolumn` unless `\pcol@com@column*` is given an optional spanning text argument which is directly processed by `\pcol@sptext`, if they appear as the first column-switching command/environment after `\begin{paracol}`. Then we re`\define` `\paracol` itself so that it will complain of illegal nesting by `\PackageError`.

```

2063 \let\@elt\pcol@defcomelt \pcol@localcommands
2064 \def\column{\pcol@com@column}%
2065 \@namedef{column*}{\@nameuse{pcol@com@column*}}%
2066 \global\let\pcol@com@column\pcol@defcolumn
2067 \global\@namedef{pcol@com@column*}{\pcol@defcolumn
2068   \@ifnextchar[%]
2069     \pcol@sptext\relax}%
2070 \def\paracol##1{\PackageError{paracol}{%
2071   Environment paracol cannot be nested.}\@eha}%

```

Ninth, we let `\output` have our output routine `\pcol@output` as its sole token, and then make `\output` request with `\penalty = \pcol@op@start` by `\pcol@invokeoutput` to invoke `\pcol@output@start` for initialization, after letting `\@elt = \relax` to make it sure that any lists can be manipulated without unexpected application of a macro to their elements.

```

2072 \output{\pcol@output}%
2073 \let\@elt\relax
2074 \pcol@invokeoutput\pcol@op@start

```

Tenth and finally, we let `\pcol@nextcol = 0` in case `\begin{paracol}` has the optional argument for spanning text, and invoke `\pcol@sptext` if it has. Otherwise, we invoke `\pcol@colpream-0` being the column preamble of the first column 0, which will be invoked by `\pcol@switchcol` if spanning text is given.

```

2075 \pcol@nextcol\z@
2076 \@ifnextchar[%]
2077   \pcol@sptext{\@nameuse{pcol@colpream0}}

```

`\pcol@paracol` The macro `\pcol@paracol` has the definition of `\paracol`, which is redefined in the macro itself, so that the only referrer `\pcol@icolumncolor` examines if it appears in `paracol`, i.e., `\pcol@paracol \neq \paracol`.

```

2078 \let\pcol@paracol\paracol
2079

```

²²⁵We can do this for other commands instead of adhering to `\let` to eliminate the exception, but the author loves to use `\let` as much as possible.

`\thecolumn` The API macro `\thecolumn` gives the value of `\pcol@currcol` to users to let them know which column they are working in so that, for example, they do some column-dependent operations.

```
2080 \def\thecolumn{\number\pcol@currcol}
2081
```

`\pcol@ignore` The macro `\pcol@ignore⟨macro⟩` is to complain that the `⟨macro⟩` being one of the followings appears in `paracol` environment.

```
\twosided, \swapcolumnnineevenpages, \noswapcolumnnineevenpages,
\footnoteplacement, \footnotelayout, \multicolumnfootnotes,
\singlecolumnfootnotes, \mergedfootnotes
```

That is, these macros, except for `\twosided`, are re`\defined` in `\pcol@zparacol` to invoke this macro with the argument identifying themselves, which is shown in the warning message given by `\PackageWarning`. As for `\twosided`, the target of the re`\definition` is `\pcol@twosided` so that its optional argument is captured before the complaint.

```
2082 \def\pcol@ignore#1{\PackageWarning{paracol}{The command \string#1 is not
2083   effective in paracol environment and thus ignored}}
2084
```

`\pcol@localcommands` The macro `\pcol@localcommands` is the list of the names of the following *environment-local* API commands (or *local commands* in short) and is solely referred to by `\pcol@zparacol`.

```
\switchcolumn   \endcolumn(*)   \nthcolumn(*)   \endnthcolumn(*)
\leftcolumn(*)  \endleftcolumn(*)  \rightcolumn(*)  \endrightcolumn(*)
\flushpage      \clearpage        \cleardoublepage
\synccounter    \syncallcounters
```

Note that we omit `\column(*)` from the list as discussed in the description of `\pcol@zparacol`.

```
2085 \def\pcol@localcommands{%
2086   \@elt{switchcolumn}%
2087   \@elt{endcolumn}\@elt{endcolumn*}%
2088   \@elt{nthcolumn}\@elt{endnthcolumn}\@elt{nthcolumn*}\@elt{endnthcolumn*}%
2089   \@elt{leftcolumn}\@elt{endleftcolumn}\@elt{leftcolumn*}\@elt{endleftcolumn*}%
2090   \@elt{rightcolumn}\@elt{endrightcolumn}%
2091   \@elt{rightcolumn*}\@elt{endrightcolumn*}%
2092   \@elt{flushpage}\@elt{clearpage}\@elt{cleardoublepage}%
2093   \@elt{synccounter}\@elt{syncallcounters}}
```

`\pcol@defcomelt` The macro `\pcol@defcomelt` is invoked solely from `\pcol@zparacol` to be applied to each element `⟨com⟩` in `\pcol@localcommands`. Two lengthy `\lets` with `\expandafters` are for doing `\let⟨com⟩=\pcol@com@.⟨com⟩` to bind the environment-local API command `\⟨com⟩` and its implementation `\pcol@com@.⟨com⟩`.

```
2094 \def\pcol@defcomelt#1{%
2095   \expandafter\let\expandafter\reserved@a\csname pcol@com@#1\endcsname
2096   \expandafter\let\csname #1\endcsname\reserved@a}
2097
```

`\dbldeferlist` As discussed in §1.8, 2015 version of L^AT_EX no longer uses `\dbldeferlist` but the macro itself is still kept in L^AT_EX. However it will be removed in future to make the first `\@cons` with it resulting in an error. Therefore, here we have its top level definition with empty duplicatedly in case of its future elimination. The macro `\end@dblfloat`, on the other hand, is replaced with a new definition in the new L^AT_EX of course, and thus we define `\pcol@end@dblfloat`

here to keep its old definition and to replace the new one in `paracol` environment as discussed in item-(1) of §1.8.

```

2098 \gdef\@dbldeferlist{}
2099 \def\pcol@end@dblfloat{%
2100   \if@twocolumn
2101     \endfloatbox
2102     \ifnum\@floatpenalty <\z@
2103       \@largefloatcheck
2104       \@cons\@dbldeferlist\@currbox
2105     \fi
2106     \ifnum \@floatpenalty =-\@Mii \@Esphack\fi
2107   \else
2108     \end@float
2109   \fi
2110 }

```

13 Column Width Setting

`\columnratio` The API macro `\columnratio{ $r_0^l, r_1^l, \dots, r_{k^l-1}^l$ }[$r_0^r, r_1^r, \dots, r_{k^r-1}^r$]` defines the column width fraction r_c^l for column c in left parallel-pages and optionally r_c^r for those in right parallel-pages. This macro and its callee `\pcol@icolumnratio` just `\globally \define` macros `\pcol@columnratioleft` and `\pcol@columnratioreft` whose bodies have the first and second arguments respectively, or commonly have the first if the second optional one is not given, so that they are given to `\pcol@setcolwidth@r` as its third argument through `\pcol@setcolumnwidth` invoked in `\pcol@zparacol`. Both of `\pcol@columnratioleft` and `\pcol@columnratioreft` are initialized to be empty so that all columns have same width and are separated by `\columnsep` as default. Note that `\pcol@columnratioleft` can be made `\let`-equal to `\relax` by the related API macro `\setcolumnwidth` so that `\pcol@setcolumnwidth` knows which of specifications given by two API macros is effective and chooses `\pcol@setcolwidth@r` or `\pcol@setcolwidth@s`.

```

2111 %% Column Width Setting
2112
2113 \def\columnratio#1{\global\let\pcol@colwidthspecleft\relax
2114   \gdef\pcol@columnratioleft{#1}%
2115   \ifnextchar[%]
2116     \pcol@icolumnratio{\gdef\pcol@columnratioreft{#1}}
2117 \def\pcol@icolumnratio[#1]{\gdef\pcol@columnratioreft{#1}}
2118 \columnratio{ }\relax
2119

```

`\setcolumnwidth` The API macro `\setcolumnwidth{ $s_0^l, s_1^l, \dots, s_{k^l-1}^l$ }[$s_0^r, s_1^r, \dots, s_{k^r-1}^r$]` defines the column width specification s_c^l for column c in left parallel-pages and optionally s_c^r for those in right parallel-pages, where each specification s_c^x has the form of $[w_c][/[g_c]]$ for width and gap specifier w_c and g_c . After `\letting` `\pcol@columnratioleft = \relax` to disable the setting by `\columnratio` and to enable that done by this macro, the macro and its callee `\pcol@isecolumnwidth` just `\globally \define` macros `\pcol@colwidthspecleft` and `\pcol@colwidthspecright` whose bodies have the first and second arguments respectively, or commonly have the first if the second optional one is not given, so that they are given to `\pcol@setcolwidth@s` as its fourth argument through `\pcol@setcolumnwidth` invoked in

`\pcol@zparacol`. Both of `\pcol@colwidthspecleft` and `\pcol@colwidthspecright` are initially undefined because the default specification is given by `\columnratio{}`.

```
2120 \def\setcolumnwidth#1{\global\let\pcol@columnratioleft\relax
2121   \gdef\pcol@colwidthspecleft{#1}%
2122   \@ifnextchar[%]
2123     \pcol@isetcolumnwidth{\gdef\pcol@colwidthspecright{#1}}
2124 \def\pcol@isetcolumnwidth[#1]{\gdef\pcol@colwidthspecright{#1}}
2125
```

`\pcol@setcolumnwidth` The macro `\pcol@setcolumnwidth` $\langle C^0 \rangle \langle C^1 \rangle \langle ratio \rangle \langle spec \rangle$ is invoked solely from `\pcol@zparacol` but can be twice, with;

$$(C^0, C^1, \langle ratio \rangle, \langle spec \rangle) = (0, C_L, \pcol@columnratioleft, \pcol@colwidthspecleft)$$

always and with;

$$(C^0, C^1, \langle ratio \rangle, \langle spec \rangle) = (C_L, C, \pcol@columnratoright, \pcol@colwidthspecright)$$

if $C_L < C$ for parallel-paging. The macro simply invokes `\pcol@setcolwidth@s` if `\pcol@columnratioleft = \relax` because `\setcolumnwidth` did so, or `\pcol@setcolwidth@r` otherwise, with all arguments given by `\pcol@zparacol`.

```
2126 \def\pcol@setcolumnwidth{%
2127   \ifx\pcol@columnratioleft\relax \let\reserved@a\pcol@setcolwidth@s
2128   \else \let\reserved@a\pcol@setcolwidth@r
2129   \fi
2130   \reserved@a}
2131
```

`\pcol@setcolwidth@r` The macro `\pcol@setcolwidth@r` $\langle C^0 \rangle \langle C^1 \rangle \langle ratio \rangle \langle spec \rangle$ is invoked solely from `\pcol@zparacol` through `\pcol@setcolumnwidth` once or twice with the arguments described in the explanation of the latter macro. The macro calculates $w_c = \pcol@columnwidth \cdot c$ for all $c \in [C^0, C^1)$, from the fractions r_0, r_1, \dots, r_{k-1} given through the third argument $\langle ratio \rangle$, which was given to `\columnratio` and then kept in `\pcol@columnratioleft` or `\pcol@columnratoright`. The macro also lets $g_c = \pcol@columnsep \cdot c = \columnsep$ for all $c \in [C^0, C^1)$.

First, we calculate $W = \text{textwidth} - (C^1 - C^0 - 1) \times \columnsep$ being the sum of w_c for all $c \in [C^0, C^1)$. Then we let $w_c = r_d W$ and $g_c = \columnsep$ for all $c \in [C^0, k')$ where $k' = \min(k, C^1 - 1)$, in the `\@for` loop scanning r_d for all $d = c - C^0 \in [0, k)$. Finally, we let $w_c = (W - \sum_{d=C^0}^{k'-1} w_d) / (C^1 - C^0 - k')$ and $g_c = \columnsep$ for all $c \in [k', C^1)$. Note that `\pcol@columnwidth \cdot c` and `\pcol@columnsep \cdot c` are macros having the integer representations of w_c and g_c with the unit `sp`.

```
2132 \def\pcol@setcolwidth@r#1#2#3#4{%
2133   \@tempcntb#2\advance\@tempcntb-#1\advance\@tempcntb\m@ne
2134   \@tempdima-\columnsep \multiply\@tempdima\@tempcntb
2135   \advance\@tempdima\textwidth \@tempdimb\@tempdima
2136   \@tempcnta#1\relax\@tempcntb#2\advance\@tempcntb\m@ne
2137   \@for\reserved@a:=#3\do{%
2138     \ifnum\@tempcnta<\@tempcntb
2139       \@tempdimc\reserved@a\@tempdima
2140       \expandafter\xdef\csname pcol@columnwidth\number\@tempcnta\endcsname{%
2141         \number\@tempdimc sp}%
2142       \global\@namedef{pcol@columnsep\number\@tempcnta}{\columnsep}%
2143       \advance\@tempdimb-\@tempdimc
```

```

2144     \advance\@tempcnta\@ne
2145     \fi}%
2146 \@tempcntb#2\advance\@tempcntb-\@tempcnta
2147 \divide\@tempdimb\@tempcntb
2148 \@whilenum\@tempcnta<#2\do{%
2149   \expandafter\xdef\csname pcol@columnwidth\number\@tempcnta\endcsname{%
2150     \number\@tempdimb sp}%
2151   \global\@namedef{pcol@columnsep\number\@tempcnta}{\columnsep}%
2152   \advance\@tempcnta\@ne}%
2153 }
2154

```

`\pcol@setcolwidth@s` The macro `\pcol@setcolwidth@s` $\langle C^0 \rangle \langle C^1 \rangle \langle ratio \rangle \langle spec \rangle$ is invoked solely from `\pcol@zparacol`
`\pcol@setcw@c` through `\pcol@setcolumnwidth` once or twice with the arguments described in the explanation
`\pcol@setcw@s` of the latter macro. The macro calculates $w_c = \pcol@columnwidth \cdot c$ for all $c \in [C^0, C^1)$ and
`\pcol@setcw@filunit` $g_c = \pcol@columnsep \cdot c$ for all $c \in [C^0, C^1 - 1)$, from the column/gap specifications s_0, s_1, \dots, s_{k-1} given through the fourth argument $\langle spec \rangle$, which was given to `\setcolumnwidth` and then kept in `\pcol@colwidthspecleft` or `\pcol@colwidthspecright`.

Each specification s_d for w_c and g_c where $c = C^0 + d$ has the form $[w'_d] [/ [g'_d]]$ to specify the natural width w'_d and g'_d and infinite stretch factor w_d^f and g_d^f of column/gap specification as follows;

$$w_d^n = \begin{cases} 0 & w'_d = \emptyset \\ 0 & w'_d = f \backslash \text{fill} \\ \text{natural}(w'_d) & \text{otherwise} \end{cases} \quad w_d^f = \begin{cases} 1 & w'_d = \emptyset \\ f & w'_d = f \backslash \text{fill} \\ \text{stretch}(w'_d) & \text{otherwise} \end{cases}$$

$$g_d^n = \begin{cases} \backslash \text{columnsep} & g'_d = \emptyset \\ 0 & g'_d = f \backslash \text{fill} \\ \text{natural}(g'_d) & \text{otherwise} \end{cases} \quad g_d^f = \begin{cases} 0 & g'_d = \emptyset \\ f & g'_d = f \backslash \text{fill} \\ \text{stretch}(g'_d) & \text{otherwise} \end{cases}$$

where $\text{natural}(x)$ is the natural width of the skip x and $\text{stretch}(x)$ is the infinite stretch factor of x . Note that any finite stretch factors or any shrink factors do not affect them, and infinite stretch units `fil`, `fill` and `filll` are not distinguished. From factors above, we determine w_c and g_c as follows;

$$W = \sum_{d=0}^{m-2} (w_d^n + g_d^n) + w_{m-1}^n$$

$$F = \sum_{d=0}^{m-2} (w_d^f + g_d^f) + w_{m-1}^f$$

$$x_c = \begin{cases} (W_T/W)x_{c-C^0} & W \geq W_T \vee F \leq 0 \\ x_{c-C^0}^n + (x_{c-C^0}^f/F)(W_T - W) & W < W_T \wedge F > 0 \end{cases} \quad (x \in w, g)$$

where $W_T = \text{textwidth}$ and $m = C^1 - C^0$.

To perform the assignments above, the macro at first invoke `\pcol@setcw@scan` $\langle C^0 \rangle \langle C^1 \rangle$
 $\{spec\}$ letting `\pcol@setcw@c = \pcol@setcw@s = \pcol@setcw@accumwd` and `\pcol@setcw@filunit = 1pt` to scan s_d for all $d \in [0, m)$ and to accumulate $W + g_{m-1}^n$ and $F + g_{m-1}^f$ in `\dimen@` and `\dimen@ii` and then subtract g_{m-1}^n and g_{m-1}^f from them to have W and F . Note that F is represented by a dimension with the unit of pt by the definition of `\pcol@setcw@filunit`. Then we invoke `\pcol@setcw@calc factors` to calculate $(W_T/W) = \pcol@setcw@scale$ and $(W_T - W)/F = \@tempdimb$. Finally we scan s_d

again but in this case we let $\backslash\text{pcol@setcw@c} = \backslash\text{pcol@setcw@set\{width\}}$, $\backslash\text{pcol@setcw@s} = \backslash\text{pcol@setcw@set\{sep\}}$ and $\backslash\text{pcol@setcw@filunit} = \backslash\text{@tempdimb} = (W_T - W)/F$ to let w_c and g_c have the values shown above.

```

2155 \def\pcol@setcolwidth@s#1#2#3#4{\begingroup
2156 \dimen@z@ \dimen@ii\z@ \def\pcol@setcw@filunit{\@ne\p@}%
2157 \let\pcol@setcw@c\pcol@setcw@accumwd \let\pcol@setcw@s\pcol@setcw@accumwd
2158 \pcol@setcw@scan#1#2{#4}%
2159 \advance\dimen@-\@tempdima \advance\dimen@ii-\@tempdimb
2160 \pcol@setcw@calc factors
2161 \def\pcol@setcw@c{\pcol@setcw@set\{width\}}%
2162 \def\pcol@setcw@s{\pcol@setcw@set\{sep\}}%
2163 \let\pcol@setcw@filunit\dimen@ii
2164 \pcol@setcw@scan#1#2{#4}%
2165 \endgroup

```

$\backslash\text{pcol@setcw@scan}$ The macro $\backslash\text{pcol@setcw@scan}\langle C^0\rangle\langle C^1\rangle\{spec\}$ is to scan first $m = C^1 - C^0$ elements in $\langle spec \rangle = s_0, s_1, \dots$ being the column/gap specifications given to $\backslash\text{setcolumnwidth}$. At first we add ‘,’ as many as m to the tail of $\langle spec \rangle$ to make it sure the resulting $\langle spec \rangle$ has m or more elements. Then we scan all elements in the extended $\langle spec \rangle$ by a $\backslash\text{@for}$ loop having many $\backslash\text{expandafter}$ but equivalent to;

```
\@for\reserved@a:=s_0,s_1,\dots\do{body}
```

In the *body* above, we invoke $\backslash\text{pcol@setcw@getspec } s_i//\@nil$ to parse s_i to have w_i^n, w_i^f to be processed by $\backslash\text{pcol@setcw@c}$ and g_i^n and g_i^f by $\backslash\text{pcol@setcw@s}$, for all $i \in [0, m)$.

```

2166 \def\pcol@setcw@scan#1#2#3{\def\reserved@a{#3}%
2167 \@tempcnta#1\relax \@whilenum\@tempcnta<#2\do{
2168   \edef\reserved@a{\reserved@a,}\advance\@tempcnta\@ne}%
2169 \@tempcnta#1\relax
2170 \expandafter\@for\expandafter\reserved@a\expandafter:\expandafter=\reserved@a
2171   \do{%
2172     \ifnum\@tempcnta<#2\relax
2173       \expandafter\pcol@setcw@getspec\reserved@a//\@nil
2174     \fi
2175     \advance\@tempcnta\@ne}}

```

$\backslash\text{pcol@setcw@getspec}$ The macro $\backslash\text{pcol@setcw@getspec}\langle w'_d\rangle/\langle g'_d\rangle/\langle garbage\rangle\@nil$ is used solely in $\backslash\text{pcol@setcw@scan}$ to parse a column/gap specification $s_d = [w'_d] [/[g'_d]]$, to extract factors w_d^n, w_d^f, g_d^n and g_d^f , and to process width factors by $\backslash\text{pcol@setcw@c}$ and gap factors by $\backslash\text{pcol@setcw@s}$. Since the macro is invoked with arguments in the form of $s_d//\@nil$, if s_d has ‘/’ in it w'_d and g'_d should have everything preceding and following the ‘/’ respectively while $\langle garbage \rangle$ should have the redundant ‘/’. Otherwise, i.e., if s_d does not have ‘/’, $w'_d = s_d$ and $g'_d = \emptyset$ while $\langle garbage \rangle = \emptyset$ ²²⁶. Therefore, we invoke $\backslash\text{pcol@setcw@getspec@i}\langle default\rangle\{x'_d\}$ twice with $(\langle default\rangle, x'_d) = (\backslash\text{fill}, w'_d)$ for a column and $(\langle default\rangle, x'_d) = (\backslash\text{columnsep}, g'_d)$ for a gap, and $\backslash\text{pcol@setcw@c}$ and $\backslash\text{pcol@setcw@s}$ after each invocation respectively.

In this macro, at first we scan all tokens in x'_d by $\backslash\text{@tfor}$ to remove all space tokens in it²²⁷. Then if x'_d after the space removal has nothing, we let $x'_d = \langle default \rangle$. Next we examine if $x'_d = f\backslash\text{fill}$ in a tricky way by temporarily $\backslash\text{letting } \backslash\text{@gtempa} = \backslash\text{relax}$, defining

²²⁶If s_d have two or more ‘/’, everything following the second one is thrown away into $\langle garbage \rangle$ together with ‘//’. Therefore we could check if s_d has at most one ‘/’ by examining $\langle garbage \rangle$ but we abandon it simply ignoring $\langle garbage \rangle$.

²²⁷A proper skip specification and “ $f\backslash\text{fill}$ ” is always proper without space tokens in them.

`\fill` as “`1pt\gdef\@gtempa{}`” and making an assignment “`\@tempskipa x'_d`”. That is, if $x'_d = f\text{fill}$, `\@tempskipa` will have $f \cdot 1\text{pt}$ being a proper dimension and `\@gtempa` is made empty. Otherwise, `\@tempskipa` should have x'_d which must be a proper skip and `\@gtempa` remains unchanged from `\relax`. Therefore, if `\@gtempa = \relax` we let `\@tempskipa = x'_d` again²²⁸. Otherwise, we invoke `\pcol@setcw@fill x'_d` to let `\@tempskipa` have 0pt plus $f\text{fil}$ where f is replaced by 1 if $f = \emptyset$.

Now `\@tempskipa` has x'_d as its natural component and may have some infinite stretch component x'_d specified explicitly or with `\fill`. Therefore, we assign `\@tempskipa` to `\@tempdima` so that it has x'_d , and then, after adding 0pt plus 1000pt minus 1000pt to `\@tempskipa` to make it sure it has both stretch and shrink components²²⁹ keeping infinite stretch factor if any, invoke `\pcol@extract@fil` giving it `\the-expansion of \@tempskipa` as the argument to let `\@tempdimb = x'_d \times \pcol@setcw@filunit`.

```

2176 \def\pcol@setcw@getspec#1/#2/#3\@nil{%
2177   \pcol@setcw@getspec@i\fill{#1}\pcol@setcw@c
2178   \pcol@setcw@getspec@i\columnsep{#2}\pcol@setcw@s}
2179 \def\pcol@setcw@getspec@i#1#2{%
2180   \def\reserved@a{}%
2181   \@tfor\reserved@b:=#2\do{\edef\reserved@a{\reserved@a\reserved@b}}
2182   \ifx\reserved@a\@empty \let\reserved@a#1\fi
2183   \let\@gtempa\relax
2184   {\def\fill{1\p@\gdef\@gtempa{}}\@tempskipa\reserved@a}%
2185   \ifx\@gtempa\relax \@tempskipa\reserved@a\relax
2186   \else \expandafter\pcol@setcw@fill\reserved@a
2187   \fi
2188   \@tempdima\@tempskipa
2189   \advance\@tempskipa0\p@\@plus\@m\p@\@minus\@m\p@\relax
2190   \expandafter\pcol@extract@fil\the\@tempskipa\@nil}
2191 \def\pcol@setcw@fill#1\fill{\def\reserved@b{#1}%
2192   \ifx\reserved@b\@empty \let\reserved@b\@ne \fi
2193   \@tempskipa0\p@\@plus\reserved@b fil\relax}
2194

```

`\pcol@setcw@accumwd` The macro `\pcol@setcw@accumwd` is made `\let`-equal to `\pcol@setcw@c` and `\pcol@setcw@s` commonly and thus invoked from `\pcol@setcw@getspec` with setting `\@tempdima = x'_d` and `\@tempdimb = x'_d \times 1\text{pt}`, where $x \in \{w, g\}$. The macro simply add them to `\dimen@` and `\dimen@ii` respectively to accumulate x'_d and x'_d in them.

```

2195 \def\pcol@setcw@accumwd{\advance\dimen@\@tempdima \advance\dimen@ii\@tempdimb}

```

`\pcol@setcw@set` The macro `\pcol@setcw@set{wors}` is made the body of `\pcol@setcw@c` with `\langle wors \rangle = width` and of `\pcol@setcw@s` with `\langle wors \rangle = sep` and thus invoked from `\pcol@setcw@getspec` with setting `\@tempdima = x'_d` and

$$(\pcol@setcw@scale, \@tempdimb) \in \{(\emptyset, (x'_d/F)(W_T - W)), (W_T/W, 0)\}$$

Therefore, we calculate $x_c = \pcol@setcw@scale \times \@tempdima + \@tempdimb$ and `\xdefine \pcol@column-\langle wors \rangle-c` to let it have the integer representation of x_c with the unit `sp`.

```

2196 \def\pcol@setcw@set#1{%
2197   \@tempdima\pcol@setcw@scale\@tempdima \advance\@tempdima\@tempdimb
2198   \expandafter\xdef\csname pcol@column#1\number\@tempcnta\endcsname{#1}

```

²²⁸Because the first assignment is done in a group.

²²⁹Almost sure because they could be -1000pt , but we ignore the possibility.

2199 \number\@tempdima sp}}
 2200

\pcol@setcw@calc@factors The macro \pcol@setcw@calc@factors is used solely in \pcol@setcolwidth@s to calculate
 \pcol@setcw@calc@f \phi_s = \pcol@setcw@scale and \phi_f = \dimen@ii as follows
 \pcol@setcw@scale

$$(\phi_s, \phi_f) = \begin{cases} (W_T/W, 0) & W \geq W_T \vee F \leq 0 \\ (1, (W_T - W)/F) & W < W_T \wedge F > 0 \end{cases}$$

where $W = \dimen@$, $F \times 1 \text{ pt} = \dimen@ii$ and $W_T = \text{textwidth}$, and $\phi_s = 1$ is represented by empty body of \pcol@setcw@scale. First we deal with the special and trivial case of $W = W_T$ to let $\phi_s = 1$ and $\phi_f = 0$ so as to avoid arithmetic error in the calculation of W_T/W . If $W \neq W_T$ on the other hand, we calculate W_T/W by \pcol@setcw@calc@f\langle W_T \rangle \langle W \rangle \langle \phi_s \rangle to have a provisional result. Then if $W < W_T$ and $F > 0$, we let $\phi_s = 1$ and invoke \pcol@setcw@calc@f again giving it $(W_T - W)$ and $F \times 1 \text{ pt}$ but throw away the result $(W_T - W)/(F \times 1 \text{ pt})$ because \@tempdimb should have $(W_T - W)/F$ which is then set into $\phi_f = \dimen@ii$. Otherwise, we keep the provisional result of ϕ_s as the final one and let $\phi_f = \dimen@ii = 0$.

The macro \pcol@setcw@calc@f\langle x \rangle \langle y \rangle \langle z \rangle calculates $z \approx x/y$ and let \@tempdimb = $Z = z \times 1 \text{ pt}$ as follows. First we find the following three parameters.

$$\begin{aligned} k_1 &= \min\{k \mid k \geq 0, x \cdot 2^k \geq 2^{13} \text{ pt}\} \\ k_2 &= \max\{k \mid y \bmod 2^k = 0\} \\ k_3 &= \min\{k \mid k \geq 0, \lceil y/2^{k_2+k} \rceil \leq 2^{15}\} \end{aligned}$$

With these parameters, we calculate $z' = \lfloor (x \cdot 2^{k_1}) / \lceil y/2^{k_2+k_3} \rceil \rfloor$ to have a good approximation of $(x/y) \cdot 2^k$ where $k = k_1 + k_2 + k_3$ without arithmetic overflow. Then if $z'/2^k \geq 2^{14}$ or in other words Z is larger than \maxdimen, we complain that by \PackageError and, in case a user dare to continue the typesetting process, we let $Z = 10000 \text{ pt}$. Otherwise, we calculate $Z = (z'/2^k) \cdot 2^{16} = z' \cdot 2^{16-k}$ to have it in \@tempdimb by $Z = z' \times 2^{16-k}$ if $k < 16$, or by $Z = z'/2^{k-16}$ otherwise. Finally we invoke \pcol@extract@pt giving it \the-representation of Z to have z .

Note that it is assured $z \leq x/y$ regardless of successfulness of the calculation and thus the scaling $\phi_s x_d^n$ and stretching $x_d^n + \phi_f x_d^f$ cannot exceed their exact value to make it also sure that $\sum_{c=C_0}^{C_1-2} (w_c + g_c) + w_{C_1-1} \leq W_T$ and thus the series of columns and column-separating gaps should not cause overflow when a page is shipped out with \hfil added to each column-separating gap for underfull avoidance.

```
2201 \def\pcol@setcw@calc@factors{%
2202   \ifdim\dimen@=\textwidth \def\pcol@setcw@scale{\dimen@ii}\z@
2203   \else
2204     \pcol@setcw@calc@f\textwidth\dimen@\pcol@setcw@scale
2205     \ifdim\dimen@<\textwidth \ifdim\dimen@ii>\z@
2206       \def\pcol@setcw@scale{}%
2207       \@tempdimc\textwidth \advance\@tempdimc-\dimen@
2208       \pcol@setcw@calc@f\@tempdimc\dimen@ii\reserved@a \dimen@ii\@tempdimb
2209     \else \dimen@ii\z@ \fi
2210     \else \dimen@ii\z@ \fi
2211   \fi}
2212 \def\pcol@setcw@calc@f#1#2#3{%
2213   \@tempdimb#1\@tempdima#2\@tempcnta\z@
2214   \ifdim\@tempdima=\z@ \@tempdima1sp\relax\fi
2215   \@whiledim\@tempdimb<8192\p@\do{%
```

```

2216 \multiply\@tempdimb\tw@ \advance\@tempcnta\@ne}%
2217 \@tempdimc\@tempdima
2218 \@whiledim\@tempdima=\@tempdimc\do{%
2219 \divide\@tempdimc\tw@ \multiply\@tempdimc\tw@
2220 \ifdim\@tempdima=\@tempdimc
2221 \divide\@tempdima\tw@ \divide\@tempdimc\tw@ \advance\@tempcnta\@ne
2222 \fi}
2223 \advance\@tempdima-1sp\relax
2224 \@whilenum\@tempdima>32768\do{\divide\@tempdima\tw@ \advance\@tempcnta\@ne}%
2225 \advance\@tempdima1sp\relax
2226 \divide\@tempdimb\@tempdima \@tempdimc\@tempdimb \@tempcntb\@tempcnta
2227 \@whilenum\@tempcntb>\z@\do{\divide\@tempdimc\tw@ \advance\@tempcntb\@m@ne}
2228 \ifnum\@tempdimc>16383\relax
2229 \PackageError{%
2230 Scaling/filling factor for column/gap width is too large.}\@eha
2231 \@tempdimb\@M\p@
2232 \else
2233 \@tempcntb\sixt@@n \advance\@tempcntb-\@tempcnta
2234 \ifnum\@tempcntb<\z@
2235 \@whilenum\@tempcntb<\z@\do{\divide\@tempdimb\tw@ \advance\@tempcntb\@ne}%
2236 \else
2237 \@whilenum\@tempcntb>\z@\do{%
2238 \multiply\@tempdimb\tw@ \advance\@tempcntb\@m@ne}%
2239 \fi
2240 \fi
2241 \expandafter\pcol@extract@pt\the\@tempdimb#3}
2242

```

`\pcol@defkw` The macro `\pcol@defkw1.0<pt>_<plus>_1.0<fil>_<minus>_1.0<garbage>\@nil` is used just once at the top level to `\define \pcol@kw@pt, \pcol@kw@plus, \pcol@kw@minus` and `\pcol@kw@fil` letting them have `<pt> = pt`, `<plus> = plus`, `<minus> = minus` and `<fil> = fil` in their body respectively but with `\catcode = 12` (other) which is used in `\the`-representation of glues. For `\pcol@kw@fil` the definition, we invoke `\pcol@defkw` giving it `\the`-representation of `\@tempskipa` letting it have `1pt plus 1fil minus 1fil` having all keywords we need to have²³⁰. The macro `\pcol@kw@pt` is used in `\pcol@extract@fil@ii<unit>\@nil` to examine if `<unit> = pt`, and in `\pcol@def@extract@pt` to `\define \pcol@extract@pt` having `pt` in its argument specification. The macros `\pcol@kw@plus` and `\pcol@kw@minus` are used only in `\pcol@def@extract@fil`, and `\pcol@kw@fil` only in `\pcol@def@extract@fil@iii`, to `\define \pcol@extract@fil` having `plus` and `minus`, and `\pcol@extract@fil@iii` having `fil`, in their argument specifications respectively.

```

2243 \@tempskipa 1\p@\@plus1fil\@minus1fil\relax
2244 \def\pcol@defkw1.0#1 #2 1.0#3 #4 1.0#5\@nil{%
2245 \def\pcol@kw@pt{#1}\def\pcol@kw@plus{#2}\def\pcol@kw@fil{#3}%
2246 \def\pcol@kw@minus{#4}}
2247 \expandafter\pcol@defkw\the\@tempskipa\@nil
2248

```

²³⁰We can do what `\pcol@defkw` does by temporarily giving `\catcode = 12` to the characters for the keywords of course, but this method is much easier.

`\pcol@def@extract@fil` The macro `\pcol@extract@fil` $\langle garbage_1 \rangle$ `_plus_` $\langle s \rangle$ `_minus_` $\langle garbage_2 \rangle$ `\@nil` is used solely in `\pcol@extract@fil` `\pcol@setcw@getspec@i` to extract the infinite stretch factor f in the stretch component s of a `\pcol@extract@fil@i` column/gap specification x'_d and to let `\@tempdimb` = $f \cdot u$ where $u = \text{\pcol@setcw@filunit} \in \{\text{1pt}, \phi_f = \text{\dimen@ii}\}$ if f exist or `\@tempdimb` = 0 otherwise. First of all, since the macro `\pcol@def@extract@fil@iii` has keywords `plus` and `minus` in `\catcode` = 12 in its argument specification, we `\define` it `\pcol@extract@fil@iii` using `\pcol@def@extract@fil`, whose body is equivalent to

```
\def\pcol@extract@fil#1_plus_#2_minus#3\@nil{\pcol@extract@fil@i#2\@nil}
```

just once at the top level. Then since s should have the form $\langle n \rangle . \langle m \rangle \langle unit \rangle$ where n and m are decimal digit sequences and $\langle unit \rangle \in \{\text{pt}, \text{fil}, \text{fill}, \text{filll}\}$, we examine if $\langle unit \rangle = \text{pt}$ or not by a tricky way in `\pcol@extract@fil@i` $\langle n \rangle . \langle m \cdot unit \rangle$ `\@nil`. That is, we do `\count@` $\langle m \cdot unit \rangle$ `\@nil` with `\afterassignment` to invoke `\pcol@extract@fil@ii` $\langle unit \rangle$ `\@nil` after m is assigned to `\count@` to capture $\langle unit \rangle$. Then if it is `pt` we let `\@tempdimb` = 0, or otherwise invoke `\pcol@extract@fil@iii` $\langle f \rangle$ `fil` $\langle garbage \rangle$ `\@nil` giving it s because it should have a postfix being `fil`, `fill` or `filll`, to have `\@tempdimb` = $f \cdot u$ finally. Note that since `\pcol@extract@fil@iii` also has the keyword `fil` in its argument specification, we `\define` it using `\pcol@def@extract@fil@iii`, whose body is equivalent to

```
\def\pcol@extract@fil@iii#1fil#2\@nil{%
  \@tempdimb\pcol@setcw@filunit\relax \@tempdimb#1\@tempdimb}
```

just once at the top level too.

```
2249 \edef\pcol@def@extract@fil{%
2250   \def\noexpand\pcol@extract@fil
2251     ##1\space\pcol@kw@plus\space##2\space\pcol@kw@minus##3\noexpand\@nil{%
2252       \noexpand\pcol@extract@fil@i##2\noexpand\@nil}}
2253 \pcol@def@extract@fil
2254 \def\pcol@extract@fil@i#1.#2\@nil{\def\reserved@a{#1.#2}%
2255   \afterassignment\pcol@extract@fil@ii\count@#2\@nil}
2256 \def\pcol@extract@fil@ii#1\@nil{\def\reserved@b{#1}%
2257   \ifx\reserved@b\pcol@kw@pt \@tempdimb\z@
2258   \else \expandafter\pcol@extract@fil@iii\reserved@a\@nil
2259   \fi}
2260 \edef\pcol@def@extract@fil@iii{%
2261   \def\noexpand\pcol@extract@fil@iii##1\pcol@kw@fil##2\noexpand\@nil{%
2262     \@tempdimb\noexpand\pcol@setcw@filunit\relax \@tempdimb##1\@tempdimb}}
2263 \pcol@def@extract@fil@iii
2264
```

`\pcol@def@extract@pt` The macro `\pcol@extract@pt` $\langle f \rangle$ `pt` $\langle scale \rangle$ is solely used in `\pcol@setcw@calcf` to extract f `\pcol@extract@pt` from a dimension in the form of $f\text{pt}$ and to let the macro $\langle scale \rangle$ have f . Since this macro has the keyword `pt` in its argument specification, we `\define` it using `\pcol@def@extract@pt`, whose body is equivalent to

```
\def\pcol@extract@pt#1pt#2{\def#2{#1}}
```

just once at the top level again.

```
2265 \edef\pcol@def@extract@pt{%
2266   \def\noexpand\pcol@extract@pt##1\pcol@kw@pt##2{\def##2{##1}}}
2267 \pcol@def@extract@pt
2268
```

14 Counter Operations

`\globalcounter` The API macro `\globalcounter{ctr}`, implemented by `\pcol@globalcounter` and also used `\pcol@globalcounter@s` in `\pcol@fnlayout@p` to globalize the counter footnote, defines that $\langle ctr \rangle$ is a global counter, `\pcol@globalcounter` and thus adds it to $\Theta^g = \text{\pcol@gcounters}$, which has page at initial. Note that we examines `\pcol@gcounters` if $\langle ctr \rangle \in \Theta^g$ prior to the addition to avoid the duplication in Θ^g . Also note that initial definition of `\pcol@gcounters` is done by `\gdef` just for consistent `\global` assignments to it. On the other hand `\globalcounter*`, implemented by `\pcol@globalcounter@s`, makes all counters kept in `\cl@ckpt` global by letting `\pcol@gcounters` have the list. Switching these two functionality is done by `\globalcounter` examining if it is followed by a `*` by `\@ifstar`.

```
2269 %% Counter Operations
2270
2271 \def\globalcounter{\@ifstar\pcol@globalcounter@s\pcol@globalcounter}
2272 \def\pcol@globalcounter@s{\global\let\pcol@gcounters\cl@ckpt}
2273 \def\pcol@globalcounter#1{%
2274   \@tempswafalse \def\reserved@a{#1}%
2275   \def\@elt##1{\def\reserved@b{##1}%
2276     \ifx\reserved@a\reserved@b \@tempswatrue \fi}%
2277   \pcol@gcounters
2278   \if@tempswa\else \@cons\pcol@gcounters{{#1}}\fi}}
2279 \gdef\pcol@gcounters{\@elt{page}}
```

`\localcounter` The API macro `\localcounter{ctr}`, also used in `\pcol@fnlayout@c` to localize the counter footnote, declares that $\langle ctr \rangle$ is a local counter, and thus removes it from Θ^g by `\pcol@removecounter` if $\langle ctr \rangle \neq \text{page}$.

```
2280 \def\localcounter#1{%
2281   \expandafter\ifx\csname c@#1\endcsname\c@page\else
2282     \pcol@removecounter\pcol@gcounters{#1}%
2283   \fi}
```

`\pcol@remctrelt` The macro `\pcol@remctrelt{\theta^g}` is invoked solely from `\pcol@zparacol` and is applied to each `\pcol@removecounter` global counter $\theta^g \in \Theta^g$ to remove it from $\Theta = \text{\pcol@counters}$ in which we have θ^l finally. `\pcol@iremctrelt` The macro also moves `\cl@.\theta^g = \zeta(\theta^g)` to `\pcol@cl@.\theta^g` to keep the list of the descendant local counters of θ^g in it, and then re\defines `\cl@.\theta^g = \pcol@stepcounter{\theta^g}` so that it is invoked on `\stepcounter{\theta^g}` to let $val_c(\theta^l) = 0$ for all $c \in [0, C)$ and $\theta^l \in \zeta(\theta^g)$, if $\theta^g \neq \text{page}$. These operations are performed by a lengthy sequence with many occurrences of `\expandafter`, `\csname` and `\endcsname` but the sequence is equivalent to the following.

```
\let\pcol@cl@.\theta^g=\cl@.\theta^g
\ifx\c@.\theta^g\c@page\else \def\cl@.\theta^g{\pcol@stepcounter{\theta^g}}\fi
```

As for the removal of θ^g from Θ , we invoke `\pcol@removecounter{\theta^l}{\theta}` giving it $\Theta' = \Theta$ and $\theta = \theta^g$. This macro, also invoked from `\localcounter{\theta^l}` with $\Theta' = \Theta^g$ and $\theta = \theta^l$, does $\Theta'' \leftarrow \Theta'$, $\Theta' = \emptyset$, and then apply `\pcol@iremctrelt{\theta^l}` to each $\theta^l \in \Theta''$ to let $\Theta' \leftarrow \Theta' \cup \{\theta^l\}$ by `\@cons` if $\theta^l \neq \theta$.

```
2284 \def\pcol@remctrelt#1{%
2285   \expandafter\let\expandafter\reserved@a\csname cl@#1\endcsname
2286   \expandafter\let\csname pcol@cl@#1\endcsname\reserved@a
2287   \expandafter\ifx\csname c@#1\endcsname\c@page\else
2288     \namedef{cl@#1}{\pcol@stepcounter{#1}}%
2289   \fi
2290   \pcol@removecounter\pcol@counters{#1}}
```

```

2291 \def\pcol@removecounter#1#2{%
2292   \def\reserved@a{#2}\let\reserved@b#1\relax \global\let#1\empty
2293   {\def\@elt{\pcol@iremctrelt#1}\reserved@b}}
2294 \def\pcol@iremctrelt#1#2{%
2295   \def\reserved@b{#2}%
2296   \ifx\reserved@a\reserved@b\else \@cons#1{#2}\fi}
2297

```

\definethecounter The API macro `\definethecounter` $\langle\theta^l\rangle\langle c\rangle\langle rep\rangle$ define the local representation $\langle rep\rangle$ for a local counter θ^l in a column c . It defines `\pcol@thectr@ θ^l .c` to have $\langle rep\rangle$ in its body.

```

2298 \def\definethecounter#1#2#3{\@namedef{\pcol@thectr@#1#2}{#3}}

```

\pcol@thectrelt The macro `\pcol@thectrelt` $\langle\theta^l\rangle$ is invoked solely in `\pcol@zparacol` and is applied to each $\theta^l \in \Theta^l$ to define its local representation of default and that of the leftmost column 0. To give unique representations `\theH θ^l` used by `hyperref`, it prepends the current column number as `\pcol@thecurrcol` to them. The macro performs a lengthy sequence with many occurrences of `\expandafter`, `\csname` and `\endcsname` but the sequence is equivalent to the following.

```

\let\pcol@thectr@ $\theta^l$ =\the $\theta^l$ 
\ifx\pcol@thectr@ $\theta^l$ .0\relax\else \let\the $\theta^l$ =\pcol@thectr@ $\theta^l$ .0 \fi \let\theH $\theta^l$ =\pcol@th

```

```

2299 \def\pcol@thectrelt#1{%
2300   \expandafter\let\expandafter\reserved@a\csname the#1\endcsname
2301   \expandafter\let\csname pcol@thectr@#1\endcsname\reserved@a
2302   \expandafter\let\expandafter\reserved@a\csname pcol@thectr@#10\endcsname
2303   \ifx\reserved@a\relax\else
2304     \expandafter\let\csname the#1\endcsname\reserved@a
2305   \fi
2306   \@ifundefined{theH#1}{-}{%
2307     \expandafter\def\csname theH#1\endcsname\expandafter\expandafter\expandafter\endcsname
2308     \expandafter\expandafter\expandafter}%
2309     \expandafter\expandafter\expandafter\pcol@thecurrcol\csname theH#1\endcsname
2310   }%
2311 }%
2312 }
2313 \def\pcol@thecurrcol{column\number\pcol@currcol.}
2314

```

\pcol@loadctrelt The macro `\pcol@loadctrelt` $\langle\theta^l\rangle\langle val_c(\theta^l)\rangle$ is invoked from `\pcol@zparacol` and `\pcol@storecounters` `synccounter` and is applied to each element $\langle\theta^l, val_c(\theta^l)\rangle \in \Theta_c$ for a column c to define a macro `\pcol@ctr@ θ^l = v(θ^l)` having $val_c(\theta^l)$ in its body for a temporary use. This macro or its redefined version is then referred to by `\pcol@cmpctrelt` $\langle\theta^l\rangle$ or `\pcol@storectrelt` $\langle\theta^l\rangle$. The latter is invoked from `\pcol@storecounters` via `\pcol@sscounters` to add $\langle\theta^l, v(\theta^l)\rangle$ to `\@gtempa` by `\@cons` to rebuild Θ_c for a column c in `\@gtempa`.

The macro `\pcol@storecounters` is invoked solely from `\pcol@synccounter` $\langle\theta\rangle$ to update a local counter θ with $val(\theta)$ for counter synchronization. That is, `\pcol@storecounters` is used to add $\langle\theta^l, v(\theta^l)\rangle$ to `\@gtempa` for all $\theta^l \in \Theta^l$ by `\pcol@sscounters` giving it `\pcol@storectrelt` as its argument, where $v(\theta^l)$ is modified if $\theta^l = \theta$ or unmodified otherwise after it is defined by `\pcol@loadctrelt`.

```

2315 \def\pcol@loadctrelt#1#2{\@namedef{\pcol@ctr@#1}{#2}}

```

```

2316 \def\pcol@storecounters{\pcol@sscounters\pcol@storectrelt}
2317 \def\pcol@storectrelt#1{\@cons\@gtempa{#{1}}{\@nameuse{pcol@ctr@#1}}}}

```

`\pcol@savecounters` The macro `\pcol@savecounters` is invoked from `\pcol@com@syncallcounters`, `\pcol@`
`\pcol@savectrelt` `stepcounter` and `\pcol@switchcol` to let Θ_c for a column c have the list of $\langle \theta^l, val_c(\theta^l) \rangle$
where $val_c(\theta^l)$ is the value of $\backslash c@-\theta^l$ to be saved in the list. It does this operation invoking
`\pcol@sscounters` giving it `\pcol@savectrelt` as its argument.

The macro `\pcol@savectrelt(\theta^l)` adds $\langle \theta^l, val_c(\theta^l) \rangle$ to `\@gtempa` by `\@cons` to rebuild Θ_c
for a column c in `\@gtempa`.

```

2318 \def\pcol@savecounters{\pcol@sscounters\pcol@savectrelt}
2319 \def\pcol@savectrelt#1{\@cons\@gtempa{#{1}}{\number\csname c@#1\endcsname}}

```

`\pcol@sscounters` The macro `\pcol@sscounters<elt>` is invoked from `\pcol@storecounters` with $\langle elt \rangle =$
`\pcol@storectrelt` or `\pcol@savecounters` with $\langle elt \rangle = \backslash pc\@s\av\ectrelt$ to build $\Theta_c =$
`\pcol@counters·c` for a column c . To do that, it lets `\@gtempa = ()` and then apply $\langle elt \rangle$ to all
 $\theta^l \in \Theta^l = \backslash pc\@c\ounters$ to have updated Θ_c in `\@gtempa`. Then finally, `\@gtempa` is moved
into Θ_c by `\xdef`²³¹.

```

2320 \def\pcol@sscounters#1{\begingroup
2321   \global\let\@gtempa\@empty
2322   \let\@elt#1\relax \pcol@counters
2323   \let\@elt\relax
2324   \expandafter\xdef\csname pcol@counters\number\pcol@currcol\endcsname{%
2325     \@gtempa}%
2326   \endgroup}
2327

```

`\pcol@cmpctrelt` The macro `\pcol@cmpctrelt(\theta)` is invoked solely from `\pcol@zparacol` and is applied to each
 $\theta \in \Theta$ to examine if $val_0(\theta) = val(\theta)$ where $val_0(\theta)$ is in `\pcol@ctr@·\theta`. If the examination fails
including due to that `\pcol@ctr@·\theta` is undefined, we add θ to the list `\@gtempa` by `\@cons`.

```

2328 \def\pcol@cmpctrelt#1{\@tempswafalse \@tempcnta\@nameuse{c@#1}%
2329   \expandafter\ifx\csname pcol@ctr@#1\endcsname\relax \@tempswatrue
2330   \else\ifnum\@nameuse{pcol@ctr@#1}=\@tempcnta\else \@tempswatrue
2331   \fi\fi
2332   \if@tempswa \@cons\@gtempa{#{1}}\fi}
2333

```

`\synccounter` The macro `\pcol@com@synccounter(\theta)`, being the implementation of the environment-local
`\pcol@com@synccounter` API macro `\synccounter`, lets $val_c(\theta) = val(\theta)$ for all $c \in [0, C)$. That is, the value of the
counter θ is *broadcasted* to all columns for the *counter synchronization* of θ . This broadcast is
done by `\pcol@synccounter` with an argument `\@elt{\theta}` so that it works only on θ .

```

2334 \def\pcol@com@synccounter#1{\pcol@synccounter{\@elt{#1}}}

```

`\pcol@synccounter` The macro `\pcol@synccounter<lst>` is invoked from `\pcol@zparacol` with $\langle lst \rangle = \backslash @gtempa$
`\pcol@syncctrelt` and from `\pcol@com@synccounter<ctr>` with $\langle lst \rangle = \backslash @elt\{ctr\}$, to let $val_c(\theta) = val(\theta)$ for all
 $c \in [0, C)$ and all θ in $\langle lst \rangle$. To do that, at first we move $\langle lst \rangle$ into `\reserved@a` in order to
make `\@gtempa` free so that it can be used in `\pcol@storecounters`. Next, for each $c \in [0, C)$,
we let $v(\theta^l) = val_c(\theta^l)$ by applying `\pcol@loadctrelt` to all $\langle \theta^l, val_c(\theta^l) \rangle \in \Theta_c$, then scan $\langle lst \rangle$
applying `\pcol@syncctrelt(\theta)` for each θ in $\langle lst \rangle$ to let $v(\theta) = val(\theta)$, and finally store all $v(\theta^l)$
back to Θ_c by `\pcol@storecounters`, where $v(\theta)$ is `\pcol@ctr@·\theta`.

```

2335 \def\pcol@synccounter#1{[%

```

²³¹It can be done by `\global\let` more efficiently but it is lengthy due to two `\expandafter`.

```

2336 \let\@elt\relax \edef\reserved@a{#1}%
2337 \pcol@currcol\z@ \@whilenum\pcol@currcol<\pcol@ncol\do{%
2338   \let\@elt\pcol@loadctrelt \@nameuse{pcol@counters\number\pcol@currcol}%
2339   \let\@elt\pcol@syncctrelt \reserved@a
2340   \pcol@storecounters
2341   \advance\pcol@currcol\@ne}}
2342 \def\pcol@syncctrelt#1{%
2343   \expandafter\edef\csname pcol@ctr@#1\endcsname{\number\@nameuse{c@#1}}
2344

```

`\syncallcounters` The macro `\pcol@com@syncallcounters`, being the implementation of the environment-local API macro `\syncallcounters`, makes all local counters in all columns have the value in the current column. That is, for each $c \in [0, C)$, we invoke `\pcol@savecounters` to let $val_c(\theta^l) = val(\theta^l)$ for all $\theta^l \in \Theta^l$.

```

2345 \def\pcol@com@syncallcounters{%
2346   \pcol@currcol\z@ \@whilenum\pcol@currcol<\pcol@ncol\do{%
2347     \pcol@savecounters \advance\pcol@currcol\@ne}}
2348

```

`\pcol@setctrelt` The macro `\pcol@setctrelt` $\langle\theta^l\rangle\langle val_c(\theta^l)\rangle$ is solely invoked from `\pcol@switchcol` to switch to a column c and is applied to each $\langle\theta^l, val_c(\theta^l)\rangle \in \Theta_c$ to let $val(\theta^l) = val_c(\theta^l)$ globally. It also define the local representation of θ^l , being `\the $\cdot\theta^l$` , to be `\pcol@thectr@ θ^l · c` if it is defined, or otherwise to be `\pcol@thectr@ θ^l` which keeps the original `\the $\cdot\theta^l$` . This local representation definition is done by a lengthy sequence with many occurrences of `\expandafter`, `\csname` and `\endcsname`, but it is equivalent to the followings.

```

\ifx\pcol@thectr@ $\theta^l$ · $c$ \relax \let\the $\cdot\theta^l$ =\pcol@thectr@ $\theta^l$ 
\else \let\the $\cdot\theta^l$ =\pcol@thectr@ $\theta^l$ · $c$ 
\fi

```

```

2349 \def\pcol@setctrelt#1#2{%
2350   \global\csname c@#1\endcsname#2\relax
2351   \expandafter\ifx\csname pcol@thectr@#1\number\pcol@currcol\endcsname\relax
2352     \expandafter\let\expandafter\reserved@a\csname pcol@thectr@#1\endcsname
2353   \else
2354     \expandafter\let\expandafter\reserved@a
2355     \csname pcol@thectr@#1\number\pcol@currcol\endcsname
2356   \fi
2357   \expandafter\let\csname the#1\endcsname\reserved@a}
2358

```

`\pcol@stepcounter` The macro `\pcol@stepcounter` $\langle\theta^g\rangle$ is invoked from `\stepcounter` $\langle\theta^g\rangle$ for a global counter θ^g because `\c1@ θ^g` is modified by `\pcol@remctrelt` so as to invoke this macro to zero-clear local counters $\theta \in \zeta(\theta^g)$. To do that, we do the followings in a group for each $c \in [0, C)$. First we apply `\pcol@stpdldeit` $\langle\theta^l\rangle\langle val_c(\theta^l)\rangle$ to each $\langle\theta^l, val_c(\theta^l)\rangle \in \Theta_c$ to let $val(\theta^l) = val_c(\theta^l)$ locally. Then we apply `\pcol@stpclelt` $\langle\theta\rangle$ to each $\theta \in \zeta(\theta^g)$ to let $val(\theta) = 0$. Finally, we invoke `\pcol@savecounters` to let $val_c(\theta^l) = val(\theta^l)$ for all $\theta^l \in \Theta^l$ to reflect the zero-clear of $\theta \in \zeta(\theta^g)$.

After the operations above, we apply `\@stpelt` $\langle\theta\rangle$ to each $\theta \in \zeta(\theta^g)$ for global zero-clearing.

```

2359 \def\pcol@stepcounter#1{\begingroup
2360   \pcol@currcol\z@ \@whilenum\pcol@currcol<\pcol@ncol\do{%
2361     \let\@elt\pcol@stpdldeit \@nameuse{pcol@counters\number\pcol@currcol}%

```

```

2362 \let\@elt\pcol@stpcl@elt \@nameuse{pcol@c1@#1}%
2363 \pcol@savecounters
2364 \advance\pcol@currcol\@ne}%
2365 \endgroup
2366 \let\@elt\@stpelt \@nameuse{pcol@c1@#1}}
2367 \def\pcol@stpldelt#1#2{\csname c@#1\endcsname#2\relax}
2368 \def\pcol@stpcl@elt#1{\csname c@#1\endcsname\z@}
2369

```

15 Column-Switching Commands and Environments

`\pcol@par` Before giving the definition of column-switching commands and environments, we define a commonly used macro `\pcol@par`, which do `\par` if necessary, i.e., we are not in vertical mode. The reason why we don't simply do `\par` is that it may have some definition different from `\@par` and thus an incautious repetition of `\par` may cause undesirable results. This macro is used in `\pcol@com@switchcolumn`, `\pcol@sptext`, `\pcol@com@endcolumn`, `\pcol@flushclear`, and `\endparacol`.

```

2370 %% Column-Switching Commands and Environments
2371
2372 \def\pcol@par{\ifvmode\else \par \fi}
2373

```

`\switchcolumn` The macro `\pcol@com@switchcolumn[d]`, being the implementation of the environment-local `\pcol@com@switchcolumn` API macro `\switchcolumn`, switches to the column d if provided through its optional argument, `\pcol@switchcolumn` or to $d = (c + 1) \bmod C$ otherwise where c is the ordinal of the current column. After making `\pcol@iswitchcolumn` it sure to be in vertical mode by `\pcol@par`, it invokes `\pcol@defcolumn` to give `\pcol@com@column(*)` their `\definitions` for occurrences not as the very first column-switching command or environment of the current `paracol` environment. Then, after calculating $d = (c + 1) \bmod C$, this macro simply invokes `\pcol@switchcol[d]` with or without the calculated d depending on the existence of the optional argument delimiter '['.

The macro `\pcol@switchcolumn[d]` lets `\pcol@nextcol = d` and confirms $0 \leq d < C$ or abort the execution by `\PackageError` if it does not hold. Then it invokes `\pcol@iswitchcolumn` if `\switchcolumn[d]` is followed by a '*', or `\pcol@switchcol` otherwise.

The macro `\pcol@iswitchcolumn[⟨text⟩]` invokes `\pcol@sptext[⟨text⟩]` if the optional argument is provided, or `\pcol@switchcol` otherwise, after letting `\ifpcol@sync = true` for explicit synchronization.

```

2374 \def\pcol@com@switchcolumn{\pcol@par
2375 \pcol@defcolumn
2376 \@tempcnta\pcol@currcol \advance\@tempcnta\@ne
2377 \ifnum\@tempcnta<\pcol@ncol\else \@tempcnta\z@ \fi
2378 \ifnextchar[%]
2379 \pcol@switchcolumn{\pcol@switchcolumn[\@tempcnta]}}
2380 \def\pcol@switchcolumn[#1]{%
2381 \pcol@nextcol#1\relax
2382 \@tempswafalse
2383 \ifnum#1<\z@ \@tempwatrue \fi
2384 \ifnum#1<\pcol@ncol\else \@tempwatrue \fi
2385 \if@tempswa
2386 \PackageError{paracol}{%
2387 Column number \number#1 must be less than \number\pcol@ncol}\@eha
2388 \pcol@nextcol\z@

```

```

2389 \fi
2390 \@ifstar\pcol@iswitchcolumn\pcol@switchcol}
2391 \def\pcol@iswitchcolumn{%
2392 \global\pcol@synctrue
2393 \@ifnextchar[%]
2394 \pcol@sptext\pcol@switchcol}
2395

```

`\pcol@sptext` The macro `\pcol@sptext[⟨text⟩]` is invoked from `\pcol@zparacol` and `\pcol@iswitchcolumn` to put a spanning text *⟨text⟩* given as the optional argument of the former or that of `\switchcolumn*` and its relative environment openers for the latter. The macro has `\long` attribute because the spanning text may have `\par`. Since the text is put in the column-0 regardless of its physical position, we let `\pcol@nextcol` have 0 after saving the target column $d = \text{\pcol@nextcol}$ in `\@tempcnta`. Then we switch to the column by `\pcol@switchcol`, after turning `\ifpcol@sync = true` to set a synchronization point above the text and `\ifpcol@sptextstart = true` to tell `\pcol@output@switch` to prepare the capture of spanning text saving the pre-spanning-text stuff.

Next, we let `\ifpcol@sptextstart = false` and `\ifpcol@sptext = true` to indicate the main vertical list contains only the spanning text and it is to be captured by `\output` routine. Then the *⟨text⟩* is put in a group in which we let `\columnwidth = \hsize = \textwidth` and `\linewidth = \textwidth - μ` with `\parshape` to indent lines by `\@totalleftmargin` if $μ > 0$, to let spanning text span across all columns reflecting the indentation in the list-like environments surrounding `paracol` if any. We also let `\col@number = 1` to ensure again that `\maketitle` produces a title without `\twocolumn` if it is in the spanning text.

Then, after invoking `\pcol@par` to ensure to be in vertical mode, we `\globalize \@svsechd` and `\@svsec` which may be defined in a lower-level sectioning command such as `\paragraph` in the spanning text so that they are properly expanded in `\everypar` inserted at the beginning of the first paragraph of the column to which we switch shortly, even when the sectioning command is used inappropriately in the spanning text. We also `\globalize \everypar` by a sequence with three `\expandafter` so that `\pcol@output@switch` for the synchronized column-switching we make shortly broadcasts it to other columns. Finally after closing the group, we let `\pcol@nextcol = d` and `\pcol@sync = true` to set another synchronization point below the spanning text and to make the captured text combined with pre-spanning-text stuff, and then invoke `\pcol@switchcol` to switch the column d .

```

2396 \long\def\pcol@sptext[#1]{%
2397 \@tempcnta\pcol@nextcol
2398 \global\pcol@synctrue \pcol@nextcol\z@
2399 \global\pcol@sptextstarttrue
2400 \pcol@switchcol
2401 \global\pcol@sptextstartfalse \global\pcol@sptexttrue
2402 \begingroup
2403 \columnwidth\textwidth \hsize\columnwidth
2404 \linewidth\columnwidth \advance\linewidth-\pcol@lrmargin
2405 \ifdim\pcol@lrmargin>\z@ \parshape\@ne\@totalleftmargin\linewidth \fi
2406 \col@number\@ne #1\pcol@par
2407 \global\let\@svsechd\@svsechd \global\let\@svsec\@svsec
2408 \expandafter\global\expandafter\everypar\expandafter{\the\everypar}%
2409 \endgroup
2410 \pcol@nextcol\@tempcnta \global\pcol@synctrue \pcol@switchcol}
2411

```

`\pcol@switchcol` The macro `\pcol@switchcol` is invoked from `\pcol@switchcolumn`, `\pcol@iswitchcolumn`,

`\pcol@sptext` and `\endparacol` to switch the column $d = \text{\pcol@nextcol}$. First, we save local counters in the current column c into Θ_c by `\pcol@savecounters`.

Next, if `\ifpcol@sync = true`, we do the followings. At first we let $V_E = \text{\pcol@@ensurevspace}$ have the natural component of `\pcol@ensurevspace` which can have a glue specified by `\ensurevspace`, so that it is referred to by `\pcol@sync` as the minimum space required below the synchronization point we are now setting. Second, we invoke `\pcol@visitallcols` temporarily turning `\ifpcol@sync = false` for column-scanning to visit all columns but current one to give T_EX's page builder the chance to break column-pages in the top page with page-wise footnotes which could have not been presented in the last visit of the columns. Third, we make an `\output` request with `\penalty = \pcol@op@switch` to invoke `\pcol@output@switch` by `\pcol@invokeoutput` with `\ifpcol@sync = true` to make synchronized switch to the column d . This invocation may result in `\ifpcol@flush = true` to mean the top page should be broken before setting the synchronization point. Therefore if so, since `\pcol@output@switch` switched to the tallest column rather than d , we put `\vfil` and `\penalty-10000` to force page break, make column-scan with `\newpage` put into each column to have some floats in the column in the new top page, and then invoke `\pcol@output@switch` again until it returns `\ifpcol@flush = false` telling us it successfully sets the synchronization point switching to the column d . Then as the last operation specific to synchronized column-switching, we invoke `\ensurevspace` with `\baselineskip` to give the default of V_E for the next synchronization.

Otherwise, i.e., if `\ifpcol@sync = false`, we simply make the `\output` request for `\pcol@output@switch` to switch to the column d .

Then we scan Θ_d applying `\pcol@setctrelt` to each $\langle \theta^l, \text{val}_d(\theta^l) \rangle \in \Theta_d$ to let $\text{val}(\theta^l) = \text{val}_d(\theta^l)$. We also scan $T = \text{\pcol@aonly}$ applying `\pcol@aonlyelt` to each $\langle t_c, c \rangle \in T$ to inhibit `\addcontentsline` to the contents file of type t_d as specified so by `\addcontentsonly(t_d)` $\langle d \rangle$. After that, we let `\@elt = \relax` to make it sure that any lists can be manipulated without unexpected application of a macro to their elements.

Finally – unless we process the implicit `\pcol@switchcol` from `\endparacol` – we invoke `\pcol@colpream-c`, where $c = -1$ if `\ifpcol@sptextstart = true` to mean the column-switching is for a spanning text, or $c = d$ otherwise.

```

2412 \def\pcol@switchcol{%
2413   \pcol@savecounters
2414   \ifpcol@sync
2415     \@tempdima\pcol@ensurevspace\relax
2416     \edef\pcol@@ensurevspace{\number\@tempdima sp\relax}%
2417     \global\pcol@syncfalse \pcol@visitallcols\@par \global\pcol@synctrue
2418     \pcol@invokeoutput\pcol@op@switch
2419     \@whiles\ifpcol@flush\fi{%
2420       \vfil \penalty-\@M
2421       \global\pcol@syncfalse \pcol@visitallcols\newpage \global\pcol@synctrue
2422       \pcol@invokeoutput\pcol@op@switch}%
2423     \ensurevspace{\baselineskip}%
2424   \else
2425     \pcol@invokeoutput\pcol@op@switch
2426   \fi
2427   \let\@elt\pcol@setctrelt
2428   \csname pcol@counters\number\pcol@currcol\endcsname
2429   \let\@elt\pcol@aonlyelt \pcol@aonly \let\@elt\relax
2430   \@ifundefined{pcol@lastcol}{%
2431     \@nameuse{pcol@colpream\ifpcol@sptextstart-1\else\number\pcol@currcol\fi}
2432   }{}}
2433
```


it in `\reserved@a` for the invocation and re`\define` it so that it will complain the illegal usage of column-switching commands/environments in the environment $\langle env \rangle$ by `\PackageError`.

```
2455 \def\pcol@switchenv#1{\let\reserved@a\switchcolumn
2456 \def\switchcolumn{\PackageError{paracol}{%
2457 Column switching commands and environments cannot be used in #1}\@eha}
2458 \reserved@a}
2459
```

`\endcolumn` The macro `\pcol@com@endcolumn` is the implementation of the environment-local API macro `\endcolumn*` `\endcolumn` to close column environment. The macro makes it sure we are in vertical mode by `\pcol@par` and `\globalize \everypar` so that it is saved in $\kappa_c(\varepsilon)$ of the current column c on the switch to another column. The macro also gives the common definition of `\pcol@com@endcolumn*` for `\endcolumn*`, `\pcol@com@endnthcolumn(*)` for `\endnthcolumn(*)`, `\pcol@com@endleftcolumn(*)` for `\endleftcolumn(*)`, and `\pcol@com@endrightcolumn(*)` for `\endrightcolumn(*)`.

```
\pcol@com@endnthcolumn* 2460 \def\pcol@com@endcolumn{\pcol@par
\endleftcolumn* 2461 \expandafter\global\expandafter\everypar\expandafter{\the\everypar}}
\endleftcolumn* 2462 \expandafter\let\csname pcol@com@endcolumn*\endcsname\pcol@com@endcolumn
\pcol@com@endleftcolumn* 2463 \let\pcol@com@endnthcolumn\pcol@com@endcolumn
\pcol@com@endleftcolumn* 2464 \expandafter\let\csname pcol@com@endnthcolumn*\endcsname\pcol@com@endcolumn
\endrightcolumn* 2465 \let\pcol@com@endleftcolumn\pcol@com@endcolumn
\endrightcolumn* 2466 \expandafter\let\csname pcol@com@endleftcolumn*\endcsname\pcol@com@endcolumn
\pcol@com@endrightcolumn* 2467 \let\pcol@com@endrightcolumn\pcol@com@endcolumn
\pcol@com@endrightcolumn* 2468 \expandafter\let\csname pcol@com@endrightcolumn*\endcsname\pcol@com@endcolumn
2469
```

`\definecolumnpreamble` The API macro `\definecolumnpreamble{c}{pream}` is to define the column preamble $\langle pream \rangle$ for the column c or that for spanning texts if $c = -1$. After assigning c to `\@tempcnta` to ensure c is a number, the macro `\pcol@colpream-c` is `\defined` to have $\langle pream \rangle$.

```
2470 \def\definecolumnpreamble#1#2{\@tempcnta#1\relax
2471 \expandafter\gdef\csname pcol@colpream\number\@tempcnta\endcsname{#2}}
2472
```

`\ensurevspace` The API macro `\ensurevspace{space}` is to declare that the synchronization point following `\pcol@ensurevspace` it must be thrown to the next page unless the page has the vertical $\langle space \rangle$ below the synchronization point. The macro makes a dummy assignment of $\langle space \rangle$ to `\@tempdima` to ensure the argument is a dimension including forced one, or in other words to raise an error if not in this macro rather than at the time $\langle space \rangle$ is *evaluated* in `\pcol@switchcol`. Then $\langle space \rangle$ is kept in `\pcol@ensurevspace` so that $\langle space \rangle$ is evaluated in `\pcol@switchcol` for the synchronization in question to pass the value to `\pcol@sync` through the macro `\pcol@@ensurevspace = V_E`, especially when it has register references, for example to `\baselineskip`. To give the default of `\pcol@ensurevspace`, we invoke `\ensurevspace` at the top level with `\baselineskip`.

```
2473 \def\ensurevspace#1{\@tempdima#1\relax \gdef\pcol@ensurevspace{#1}}
2474 \ensurevspace{\baselineskip}
2475
```

16 Disabling `\addcontentsline`

`\addcontentsonly` The API macro `\addcontentsonly{t}{c}` makes the type t contents file written by commands appearing only in the column c . The macro simply add the pair $\langle t, c \rangle$ to the list

$T = \text{\pcol@aconly}$ being empty at initial, after confirming we know the type t , one of `toc`, `lof` and `lot` so far, by the fact `\pcol@ac@def·t` is defined, or abort execution by `\PackageError`.

```

2476 %% Disabling \addcontentsline
2477
2478 \def\addcontentsonly#1#2{%
2479   \ifundefined{pcol@ac@def@#1}
2480     {\PackageError{paracol}{Unknown contents type #1}\@eha}\relax
2481   \cons\pcol@aconly{#1}{#2}}
2482 \gdef\pcol@aconly{}
2483

```

`\pcol@aconlyelt` The macro `\pcol@aconlyelt⟨ t_d ⟩⟨ d ⟩` is invoked solely in `\pcol@switchcol` for the column-switching to column c , and is applied to each $\langle t_d, d \rangle \in T$ to enable `\addcontentsline` for t_d if $d = c$ by the invocation of `\pcol@ac@def· t_d` with an argument `enable`, or to disable if $d \neq c$ with `disable`.

```

2484 \def\pcol@aconlyelt#1#2{%
2485   \ifnum#2=\pcol@currcol \@nameuse{pcol@ac@def@#1}{enable}%
2486   \else \@nameuse{pcol@ac@def@#1}{disable}%
2487   \fi}

```

`\pcol@gobblethree` The macro `\pcol@gobblethree⟨ $file$ ⟩⟨ sec ⟩⟨ $entry$ ⟩` is used in `\pcol@ac@disable@toc` and `\pcol@ac@caption` to make `\addcontentsline` `\let`-equal to this macro, which does nothing but discarding three arguments, for disabling. The macro `\pcol@addcontentsline` is the L^AT_EX's original `\addcontentsline` and is used in the macros mentioned above to let `\addcontentsline` act as original.

```

2488 \def\pcol@gobblethree#1#2#3{}
2489 \let\pcol@addcontentsline\addcontentsline
2490

```

`\pcol@ac@def@toc` The macro `\pcol@ac@def@toc⟨ $eord$ ⟩` is invoked solely in `\pcol@aconlyelt{toc}⟨ c ⟩` to enable or disable `\addcontentsline` according to $\langle eord \rangle$ by making `\@sect` `\let`-equal to `\pcol@ac@enable@toc` or `\pcol@ac@disable@toc` respectively. The macro `\pcol@ac@disable@toc⟨ a_1 ⟩⟨ a_2 ⟩⟨ a_3 ⟩⟨ a_4 ⟩⟨ a_5 ⟩⟨ a_6 ⟩[⟨ a_7 ⟩]⟨ a_8 ⟩` at first disables `\addcontentsline` by making it `\let`-equal to `\pcol@gobblethree`, then invokes the original `\@sect` saved in `\pcol@ac@enable@toc` giving it all arguments a_1 to a_8 , and finally enables it by making it `\let`-equal to `\pcol@addcontentsline`. Note that the argument a_7 is surrounded by `{` and `}` on the invocation of `\@sect` to conceal ‘`]`’ in a_7 .

```

2491 \def\pcol@ac@def@toc#1{%
2492   \expandafter\let\expandafter\@sect\csname pcol@ac@#1@toc\endcsname}
2493 \let\pcol@ac@enable@toc\@sect
2494 \def\pcol@ac@disable@toc#1#2#3#4#5#6[#7]#8{%
2495   \let\addcontentsline\pcol@gobblethree
2496   \pcol@ac@enable@toc{#1}{#2}{#3}{#4}{#5}{#6}[#7]{#8}%
2497   \let\addcontentsline\pcol@addcontentsline}
2498

```

`\pcol@ac@def@lof` The macro `\pcol@ac@def@lof⟨ $eord$ ⟩` and `\pcol@ac@def@lot⟨ $eord$ ⟩` are invoked solely in `\pcol@aconlyelt⟨ t ⟩⟨ c ⟩` when t is `lof` or `lot` respectively. They invoke `\pcol@ac@caption@enable⟨ t ⟩` or `\pcol@ac@caption@disable⟨ t ⟩` according to $\langle eord \rangle$, and then these macros invoke `\pcol@ac@caption@def⟨ s ⟩⟨ t ⟩` where $s = \text{\@tempswatru}$ or $s = \text{\@tempswafalse}$ respectively

```

\pcol@ac@caption@def
\pcol@ac@caption@if@lof
\pcol@ac@caption@if@lot

```

to let `\@caption = \pcol@ac@caption` and `\pcol@ac@caption@if@t = s` which are `\let`-equal to `\@tempswatru` in default. That is, `\pcol@ac@caption@if@t` lets `\if@tempswa = true` iff `\addcontentsline` for `t` is to be enable.

```

2499 \def\pcol@ac@def@lof#1{\@nameuse{pcol@ac@caption#1}{lof}}
2500 \def\pcol@ac@def@lot#1{\@nameuse{pcol@ac@caption#1}{lot}}
2501 \def\pcol@ac@caption@enable{\pcol@ac@caption@def\@tempswatru}
2502 \def\pcol@ac@caption@disable{\pcol@ac@caption@def\@tempswafalse}
2503 \def\pcol@ac@caption@def#1#2{\let\@caption\pcol@ac@caption
2504 \expandafter\let\csname pcol@ac@caption@if@#2\endcsname#1}
2505 \let\pcol@ac@caption@if@lof\@tempswatru
2506 \let\pcol@ac@caption@if@lot\@tempswatru

```

`\pcol@ac@caption` The macro `\pcol@ac@caption<type>[[<lap>]]<cap>` is made `\let`-equal to `\@caption` by `\pcol@ac@caption@def` to do what `\@caption` do but with enabling/disabling `\addcontentsline`. At first, it invokes `\pcol@ac@caption@if@t` where `t = \ext@<type>` to let `\if@tempswa` be `true` or `false` according to the enable/disable status of `t`. Then, after letting `\addcontentsline = \pcol@gobblethree` for disabling if `false`, we invoke `\pcol@ac@caption@latex`, being the L^AT_EX's original `\@caption`, giving all three arguments of `\pcol@ac@caption` itself surrounding `<lap>` with `{` and `}` for the concealment of `']`. Finally, we let `\addcontentsline = \pcol@addcontentsline` so that other macros uses it with its original definition.

```

2507 \long\def\pcol@ac@caption#1[#2]#3{%
2508 \@nameuse{pcol@ac@caption@if@\@nameuse{ext@#1}}%
2509 \if@tempswa\else \let\addcontentsline\pcol@gobblethree \fi
2510 \pcol@ac@caption@latex{#1}[#2][#3}%
2511 \let\addcontentsline\pcol@addcontentsline}
2512 \let\pcol@ac@caption@latex\@caption
2513

```

17 Page Flushing Commands

`\flushpage` The macros `\pcol@com@flushpage`, `\pcol@com@clearpage` and `\pcol@com@cleardoublepage` are the implementations of environment-local API macro `\flushpage`, `\clearpage` and `\cleardoublepage` respectively. The first two have a common structure in which we at first invoke `\pcol@flushclear` for column-scan and pre-flushing column height check, and then make an `\output` request by `\pcol@invokeoutput` with `\penalty` being `\pcol@op@flush` or `\pcol@op@clear` according to the commands. On the other hand the last one simply invokes `\pcol@com@clearpage` unconditionally, and then `\pcol@com@flushpage`²³² if two-sided paging is enabled by `\if@twoside = true`, we are in an even-numbered page, and `\ifpcol@paired = false` to mean we are not doing non-paired parallel-paging.

```

2514 %% Page Flushing Commands
2515
2516 \def\pcol@com@flushpage{\pcol@flushclear\voidb@x
2517 \pcol@invokeoutput\pcol@op@flush}
2518 \def\pcol@com@clearpage{\pcol@flushclear\voidb@x
2519 \pcol@invokeoutput\pcol@op@clear}
2520 \def\pcol@com@cleardoublepage{\pcol@com@clearpage
2521 \if@twoside \ifodd\c@page\else \ifpcol@paired\else \pcol@com@flushpage
2522 \fi\fi\fi}

```

²³²Unlike L^AT_EX's `\cleardoublepage`, it is unnecessary to put an empty `\hbox` before `\flushpage` because it is active even at the top of a page.

`\pcol@flushclear` The macro `\pcol@flushclear⟨box⟩`, invoked from `\pcol@com@flushpage`, `\pcol@com@clearpage` and `\endparacol`, performs column-scan and pre-flushing column height check prior to page flushing or environment closing. After confirming we are in vertical mode by `\pcol@par` and letting $d = \text{\pcol@nextcol}$ be $c = \text{\pcol@currcol}$ to stay in c , we invoke `\pcol@visitalcols` for column-scan to give T_EX's page builder the chance to break the top page prior to flushing it.

Then we repeat pre-flushing column height check invoking `\pcol@output@switch` through `\pcol@invokeoutput` with `\penalty = \pcol@op@switch` and `\ifpcol@clear = \ifpcol@sync = true` until the special `\output` routine finishes with `\ifpcol@flush = false` and `⟨box⟩ = ⊥`, where `⟨box⟩ = Φ = \pcol@topfnote` if this macro is invoked from `\endparacol` with non-merged page-wise footnote typesetting. In the repetition, we put `\vfil` and `\penalty-10000` to force page break into the tallest column, temporarily turning `\ifpcol@lastpage = false` using `\ifpcol@lastpagesave` because the broken page is not last one, each time the pre-flushing column height check tells us to do it. That is, we repeat the check while we have too tall columns due to page-wise footnotes or, when closing `paracol` environment, deferred non-merged page-wise footnotes.

Finally we let `\ifpcol@clear` have its default setting, i.e., *false*.

```

2523 \def\pcol@flushclear#1{\pcol@par
2524   \pcol@nextcol\pcol@currcol
2525   \pcol@visitalcols\@par
2526   \pcol@cleartrue \global\pcol@synctrue
2527   \ifpcol@lastpage \pcol@lastpagesavetrue \else \pcol@lastpagesavefalse \fi
2528   \pcol@invokeoutput\pcol@op@switch \ifvoid#1\else \global\pcol@flushtrue \fi
2529   \@whiles\ifpcol@flush\fi{%
2530     \pcol@lastpagefalse
2531     \vfil \penalty-\@M \pcol@cleartrue \global\pcol@synctrue
2532     \ifpcol@lastpagesave \pcol@lastpagetrue \fi
2533     \pcol@invokeoutput\pcol@op@switch
2534     \ifvoid#1\else \global\pcol@flushtrue \fi}%
2535   \pcol@clearfalse}
2536

```

18 Commands for Footnotes

`\footnoteplacement` The API macro `\footnoteplacement{l}` is to determine that footnotes are column-wise ($l = c$), page-wise without merging ($l = p$), or merged and page-wise ($l = m$). The macro examines if `\pcol@fnlayout@c` $l \in \{c, p, m\}$ by the existence of the corresponding macro `\pcol@fnlayout@l` and invokes it, or `\pcol@fnlayout@p` complains if not by `\PackageError`.

`\pcol@fnlayout@c` The macros `\pcol@fnlayout@c`, `\pcol@fnlayout@p` and `\pcol@fnlayout@m` turn switches `\multicolumnfootnotes` $f_s = \text{\ifpcol@scfnote}$, `\singlecolumnfootnotes` $f_m = \text{\ifpcol@mgfnote}$ and `\mergedfootnotes` $f_a = \text{\ifpcol@fncounteradjustment}$ and make the counter footnote global or local as follows.

l	f_s	f_m	f_a	footnote
c	false	false	false	local
p	true	false	true	global
m	true	true	true	global

Note that turning `\ifpcol@fncounteradjustment` is done by `\fncounteradjustment` (*true*) or `\nofncounteradjustment` (*false*). Also note that the setting of `\ifpcol@fncounteradjustment` and the globalization/localization of `footnote` are just to give defaults and thus can be overridden by API macros giving non-default settings. Another remark is

that backward-compatible macros `\multicolumnfootnotes`, `\singlecolumnfootnotes` and `\mergedfootnotes` are `\let`-equal to `\pcol@fnlayout@c`, `\pcol@fnlayout@p` and `\pcol@fnlayout@m` respectively. If the deprecated `\footnotelayout` is undefined `\let` it equal to `\footnoteplacement` and add a hook to undefine it if `footmisc` gets loaded; print an informational message in any case where our `\footnotelayout` is unavailable.

```

2537 %% Commands for Footnotes
2538
2539 \def\footnoteplacement#1{\@ifundefined{pcol@fnlayout@#1}%
2540 {\PackageError{paracol}{Unknown footnote layout specifier #1}}%
2541 {\@nameuse{pcol@fnlayout@#1}}}
2542 \def\pcol@fnlayout@c{\global\pcol@scfnotefalse \global\pcol@mgfnotefalse
2543 \localcounter{footnote}\nofcounteradjustment}
2544 \def\pcol@fnlayout@p{\global\pcol@scfnotetru e \global\pcol@mgfnotefalse
2545 \globalcounter{footnote}\fncounteradjustment}
2546 \def\pcol@fnlayout@m{\pcol@fnlayout@p\global\pcol@mgfnotetru e}
2547
2548 \let\multicolumnfootnotes\pcol@fnlayout@c
2549 \let\singlecolumnfootnotes\pcol@fnlayout@p
2550 \let\mergedfootnotes\pcol@fnlayout@m
2551 \@ifundefined{footnotelayout}{%
2552 \let\footnotelayout\footnoteplacement
2553 \AddToHook{package/footmisc/before}{\let\footnotelayout\relax}%
2554 }{}
2555 \AddToHook{package/footmisc/before}{\PackageNoteNoLine{paracol}{%
2556 With footmisc loaded, paracol's
2557 \string\footnotelayout\space will be unavailable. Please use the
2558 equivalent \string\footnoteplacement\space instead.\MessageBreak
2559 Also note, that paracol's and footmisc's footnote handling may crash%
2560 }}
2561

```

`\@footnotetext` The macro `\pcol@fntext{text}` is our own version of L^AT_EX's `\@footnotetext` used in `\pcol@fntext` `\footnote` and `\footnotetext` to `\insert` the footnote `<text>` through `\footins`. Since the `\pcol@fntexttop` original and our own are made `\let`-equal by `\pcol@zparacol`, our own is active throughout `\pcol@fntexttother` the environment. The customization is done to examine if the footnote should be deferred and to encapsulate the footnote in a `\vbox`.

The deferred footnote insertion is in effect if the footnote typesetting is page-wise and `\footnote` or `\footnotetext` appears in a page $p < p_t$. If so, we put the footnote `<text>` encapsulated in a `\vbox` by `\pcol@fntextbody` to the tail of $\Phi = \text{\pcol@topfnotes}$ with `\penalty\interlinepenalty` preceding it for the split in `\pcol@deferredfootins`, using `\pcol@fntexttother{text}` whose sole user is this macro. Note that the decision of deferring is done based on $p = \text{\pcol@page}$ which could be less than that of the page in which the footnoted text appears because the paragraph having the text will have a page break before the text. Therefore, p for the footnote can be p_t , but this misjudgment will not cause problems because the footnote will eventually be put in p_t through Φ when the page break occurs.

Otherwise the footnote `<text>` is processed by `\pcol@fntexttop{text}`, also used solely in this macro, to `\insert` it through `\footins` as usual but after the encapsulation by `\pcol@fntextbody` and with `\penalty\interlinepenalty` following it to allow T_EX's page builder to split footnotes.

Note that `\pcol@fntexttop` and `\pcol@fntexttother` have `\long` property because `<text>` may have two or more paragraphs.

```

2562 \def\pcol@fntext{%

```

```

2563 \let\reserved@a\pcol@fntexttop
2564 \ifpcol@scfnote \ifnum\pcol@page<\pcol@toppage
2565   \let\reserved@a\pcol@fntextother
2566 \fi\fi
2567 \reserved@a}
2568 \long\def\pcol@fntexttop#1{%
2569   \pcol@Logfn{\pcol@fntexttop{\@thefnmark}}%
2570   \insert\footins{\pcol@fntextbody{#1}\penalty\interlinepenalty}}
2571 \long\def\pcol@fntextother#1{%
2572   \global\setbox\pcol@topfnotes\vbox{\unvbox\pcol@topfnotes
2573     \penalty\interlinepenalty\pcol@fntextbody{#1}}}

```

`\pcol@fntextbody` The macro `\pcol@fntextbody{text}`, invoked from `\pcol@fntexttop` and `\pcol@fntextother`, encapsulates the footnote *text* in a `\vbox` whose height is $h_{\max} = \text{textheight} - \text{skip} \text{footins}$ at tallest. The encapsulation is to inhibit page breaks in a footnote because the split by the break will make some skips and other items eliminated causing a weird result when split portions are *joined*. The height capping is thus required to find a page in which the footnote resides.

The macro at first does operations done in L^AT_EX's `\@footnotetext` to put *text* in `\@tempboxa` but with one exception that `\hsize = \textwidth` rather than `\columnwidth` when page-wise footnote typesetting is in effect. Note that this part is blindly copied from the original though it should be meaningless to set `\interlinepenalty`, `\splittopskip`, `\splitmaxdepth` and `\floatingpenalty` because *text* is encapsulated.

Then the height-plus-depth of the box is compared with h_{\max} and, if it exceeds the limit, the height of the box is set h_{\max} , the footnote is made followed by a `\vss` to avoid overfull, and a warning message of too tall is put by `\PackageWarning`. Finally, the box is put into `\footins` or Φ by the invoker of this macro.

```

2574 \long\def\pcol@fntextbody#1{\setbox\@tempboxa\vbox{%
2575   \reset@font\footnotesize
2576   \interlinepenalty\interfootnotelinepenalty
2577   \splittopskip\footnotesep
2578   \splitmaxdepth \dp\strutbox \floatingpenalty \MM
2579   \hsize \ifpcol@scfnote \textwidth \else \columnwidth \fi
2580   \def\@currentcounter{footnote}%
2581   \protected@edef\@currentlabel{%
2582     \csname p@footnote\endcsname\@thefnmark
2583   }%
2584   \color@begingroup
2585     \makefntext{%
2586       \rule\z@\footnotesep\ignorespaces#1\@finalstrut\strutbox}%
2587   \par
2588   \color@endgroup}%
2589 \@tempdima\ht\@tempboxa \advance\@tempdima\dp\@tempboxa
2590 \@tempdimb\textheight \advance\@tempdimb-\skip\footins
2591 \ifdim\@tempdima>\@tempdimb
2592   \setbox\@tempboxa\vbox to\@tempdimb{\unvbox\@tempboxa\vss}%
2593   \PackageWarning{paracol}{Too tall footnote}%
2594 \fi
2595 \box\@tempboxa}
2596

```

`\fncounteradjustment` The API macros `\fncounteradjustment` and `\nofncounteradjustment` turns `\ifpcol@nofncounteradjustment` `fncounteradjustment` *true* or *false*, to enable or disable the footnote counter adjustment

letting $\c@footnote = b_f + n_f$ in `\end{paracol}`, respectively. After the definition we disable the adjustment to give the default setting.

```
2597 \def\fncounteradjustment{\global\pcol@fncounteradjustmenttrue}
2598 \def\nofncounteradjustment{\global\pcol@fncounteradjustmentfalse}
2599 \nofncounteradjustment
2600
```

`\pcol@footnoterule` The macros `\pcol@footnoterule`, `\pcol@@footnote`, `\pcol@@footnotemark` and `\pcol@@footnotetext` are to keep the original definitions of `\footnoterule`, `\footnote`, `\footnotemark` and `\footnotetext` in them, respectively, so that we define our own versions with references to the originals.

```
2601 \let\pcol@footnoterule\footnoterule
2602 \let\pcol@@footnote\footnote
2603 \let\pcol@@footnotemark\footnotemark
2604 \let\pcol@@footnotetext\footnotetext
```

`\footnote` The macros `\pcol@footnote` and `\pcol@footnotemark` are the implementations of our own versions of `\footnote` and `\footnotemark` which are made `\let`-equal to them by `\pcol@zparacol`, respectively. The reasons why we need to have our own are two-fold; to have starred version of them; and to maintain $n_f = \pcol@nfootnotes$ for the footnote counter adjustment.

`\pcol@iffootnotemark` The implementations of the starred versions `\footnote*[num]{text}` and `\footnotemark*[num]` have common structure in which we invoke `\pcol@adjustfnctr<macro>[num]` if `*` is given, to let `\c@footnote` have the number relative to $b_f = \pcol@footnotebase$ or to itself. Then the macros `\pcol@iffootnote` or `\pcol@iffootnotemark` are invoked from `\pcol@adjustfnctr` or the else-part of `\@ifstar` to perform the operations common to both cases with and without `*`, i.e., invoking the original version `\pcol@@footnote` or `\pcol@@footnotemark` after incrementing n_f . One caution is that `<macro> = \pcol@iffootnote` for `\footnote`, but `<macro> = {\pcol@iffootnotemark\relax}` for `\footnotemark` so that `\@ifnextchar` in `\pcol@@footnotemark` invoked from `\pcol@iffootnotemark` eats `\relax` to terminate space skipping and thus spaces following `[num]` are kept.

```
2605 \def\pcol@footnote{\@ifstar{\pcol@adjustfnctr\pcol@iffootnote}\pcol@iffootnote}
2606 \def\pcol@iffootnote{\global\advance\pcol@nfootnotes\@ne \pcol@@footnote}
2607 \def\pcol@footnotemark{\@ifstar
2608   {\pcol@adjustfnctr\pcol@iffootnotemark\relax}}%
2609   \pcol@iffootnotemark}
2610 \def\pcol@iffootnotemark{\global\advance\pcol@nfootnotes\@ne
2611   \pcol@@footnotemark}
```

`\pcol@adjustfnctr` The macro `\pcol@adjustfnctr<macro>[num]`, invoked from the then-part of `\@ifstar` in `\pcol@iffootnote` and `\pcol@iffootnotemark`, calculates the number to be set into `\c@footnote` by `\pcol@calcfnctr<num>\@nil` after processing the optional argument `<num>` by `\pcol@iadjustfnctr` with default `+1`, and then invoke `<macro>` being `\pcol@iffootnote` or `\pcol@iffootnotemark\relax`. Since `\pcol@calcfnctr` returns the number `\c@footnote` should have and the counter is incremented by `\stepcounter` in `\pcol@@footnote` or `\pcol@@footnotemark`, we decrement the counter prior to invoke `<macro>`.

The macro `\pcol@calcfnctr<num>\@nil`, also invoked from `\pcol@iffootnotetext`, calculate m specified by `<num>` as follows, where $f = \c@footnote$, to return it through `\@tempcnta`.

$$m = \begin{cases} f + k & \langle num \rangle = +k \\ f - k & \langle num \rangle = -k \\ b_f + k & \langle num \rangle = k \end{cases}$$


```

2612 \def\pcol@adjustfnctr#1{\@ifnextchar [%]
2613   {\pcol@iadjustfnctr{#1}}{\pcol@iadjustfnctr{#1}[+1]}}
2614 \def\pcol@iadjustfnctr#1[#2]{\pcol@calcfnctr#2\@nil
2615   \global\c@footnote\@tempcnta \global\advance\c@footnote\m@ne#1}
2616 \def\pcol@calcfnctr#1#2\@nil{\@tempcnta\c@footnote
2617   \def\reserved@a{#1}\def\reserved@b{+}%
2618   \ifx\reserved@a\reserved@b \advance\@tempcnta#2\relax
2619   \else \def\reserved@b{-}%
2620   \ifx\reserved@a\reserved@b \advance\@tempcnta-#2\relax
2621   \else \@tempcnta\pcol@footnotebase \advance\@tempcnta#1#2\relax
2622   \fi\fi}

```

`\footnotetext` The macros `\pcol@footnotetext` is the implementation of our own versions of `\footnotetext` which is made `\let`-equal to it by `\pcol@zparacol`. The reasons why we need to have our own is to have the starred version. That is, if `*` is not given, we simply invoke the original version `\pcol@@footnotetext`. Otherwise we invoke `\pcol@iffootnotetext` which then examines if the optional argument `[num]` is presented. If so, we invoke `\pcol@iiffootnotetext` in which `\pcol@calcfnctr<num>\@nil` is invoked to have the value `m` being the footnote ordinal with which we invoke `\pcol@@footnotetext[m]` with two `\expandafters` to extract `m` from `\@tempcnta`. Otherwise, i.e., `[num]` is not given, we increment `\c@footnote` by `\stepcounter` before invoking `\pcol@@footnotetext`.

```

2623 \def\pcol@footnotetext{\@ifstar\pcol@iffootnotetext\pcol@@footnotetext}
2624 \def\pcol@iffootnotetext{\@ifnextchar [%]
2625   \pcol@iiffootnotetext{\stepcounter{footnote}\pcol@@footnotetext}}
2626 \def\pcol@iiffootnotetext[#1]{\pcol@calcfnctr#1\@nil
2627   \expandafter\pcol@@footnotetext\expandafter [\number\@tempcnta]}
2628

```

19 Commands for Marginal Notes

`\marginpar` The API macro `\marginnote[left]{right}[voffset]` given by the package `marginnote` is emulated using `\marginpar[left]{right}` and `\pcol@addmarginpar` in `\output` routine. The basic mechanism is to pass the vertical offset `<voffset>` to `\pcol@addmarginpar` through `\dimen` where `b` is the `\insert` to carry `<left>`. The offset passing is implemented as follows.

- `\marginpar` is made `\let`-equal to our own version `\pcol@marginpar` in `\pcol@zparacol` so that it `\let` the macro `\pcol@mparoffset` be `\z@` and then invoke L^AT_EX's original version kept in `\pcol@@marginpar`, because the marginal note given by `\marginpar` will not be shifted.
- The internal macro `\@mn@@marginnote[left]{right}[voffset]` defined in `marginnote` is made `\let`-equal to our own version `\pcol@marginnote` in `\pcol@zparacol` so that it `\defines` `\pcol@mparoffset` to have `<voffset>` and then invoke `\pcol@@marginpar[left]{right}` for the emulation. In the invocation, `marginnote`'s typesetting macros `\marginfont`, `\raggedleftmarginnote` and `\raggedrightmarginnote` are attached to `<left>` and `<right>`.
- L^AT_EX's internal macro `\@xympar` for the last operations of `\marginpar` is made `\let`-equal to our own version in `\pcol@zparacol` so that it assigns the offset in `\pcol@mparoffset` to `\dimen\@marbox` for `<left>`, if `\@floatpenalty < 0` to mean other macros for `\marginpar` have not detected any errors.

In addition, we raise a warning that `\marginnote` is emulated by `\pcol@mn@warning`, which is made `\let`-equal to `\relax` in the caller `\pcol@marginnote` after the invocation so that the warning message is put just once.

```

2629 %% Commands for Marginal Notes
2630
2631 \def\pcol@marginpar{\let\pcol@mparoffset\z@ \pcol@marginpar}
2632 \long\def\pcol@marginnote[#1]#2[#3]{\endgroup
2633 \pcol@mn@warning \global\let\pcol@mn@warning\relax
2634 \def\pcol@mparoffset{#3}%
2635 \pcol@marginpar[\marginfont\raggedleftmarginnote#1]%
2636 \marginfont\raggedrightmarginnote#2}}
2637 \def\pcol@mn@warning{%
2638 \PackageWarning{paracol}{\string\marginnote\space is emulated by
2639 \string\marginpar.}}
2640 \def\pcol@xypar{%
2641 \ifnum\@floatpenalty<\z@ \global\dimen\@marbox\pcol@mparoffset\relax \fi
2642 \pcol@xypar}
2643

```

20 Two-Sided Typesetting

`\twosided` The API macro `\twosided[T]` where $T = t_1 t_2 \dots$ is to enable/disable two-sided paging with `\pcol@twosided` `\if@twoside = true/false` ($p \in/\notin T$), two-sided column-swapping with `\ifpcol@swapcolumn` `\pcol@twosided@p = true/false` ($c \in/\notin T$), two-sided marginal note placement with `\ifpcol@swapmarginpar = \pcol@twosided@c true/false` ($m \in/\notin T$), and/or two-sided background painting with `\ifpcol@bg@swap = \pcol@twosided@m true/false` ($b \in/\notin T$) individually, or to enable all of them as a whole when the optional `\pcol@twosided@b` argument is not given.

`\swapcolumninevenpages` The macro invokes `\pcol@twosided` with the optional argument T if provided, or with `\noswapcolumninevenpages` $T = \text{pcmb}$ otherwise to enable all features. Then `\pcol@twosided` at first turns all the switches *false* and then scans all non-space tokens $t \in T$ invoking `\pcol@twosided@t` if it is defined and thus $t \in \{p, c, m, b\}$ to turn the corresponding switch *true*, or complains that the feature t is unknown.

Note that backward-compatible API macros `\swapcolumninevenpages` and `\noswapcolumninevenpages` are still available to turn `\ifpcol@swapcolumn` *true* and *false* respectively.

```

2644 %% Column Swapping
2645
2646 \def\twosided{\@ifnextchar[%]
2647 {\pcol@twosided}{\pcol@twosided[pcmb]}}
2648 \def\pcol@twosided[#1]{%
2649 \global\@twosidefalse \global\pcol@swapcolumnfalse
2650 \global\pcol@swapmarginparfalse \global\pcol@bg@swapfalse
2651 \tfor\reserved@a:=#1\do{%
2652 \@ifundefined{pcol@twosided@reserved@a}%
2653 {\PackageError{paracol}{Unknown two-sideding feature \reserved@a}}%
2654 {\@nameuse{pcol@twosided@reserved@a}}}}
2655 \def\pcol@twosided@p{\global\@twosidetrue}
2656 \def\pcol@twosided@c{\global\pcol@swapcolumntrue}
2657 \def\pcol@twosided@m{\global\pcol@swapmarginpartrue}
2658 \def\pcol@twosided@b{\global\pcol@bg@swaptrue}
2659
2660 \def\swapcolumninevenpages{\global\pcol@swapcolumntrue}

```

```
2661 \def\noswapcolumninevenpages{\global\pcol@swapcolumnfalse}
2662
```

`\pcol@swapcolumn` The macro `\pcol@swapcolumn⟨ c_1 ⟩⟨ c_2 ⟩⟨ C^0 ⟩⟨ C^1 ⟩` converts the column ordinal c or position c' in the `\count` register²³³ c_1 to the position or ordinal to set it in the `\count` register c_2 , for a parallel-page having columns $c \in [C^0, C^1)$. That is, we let $c_2 = (C^1 - 1) - (c_1 - C^0)$ if `\ifpcol@swapcolumn = true` to mean the column-swapping is in effect and `\c@page` is even, while $c_2 = c_1$ otherwise. We also let $c^g = \text{\pcol@colsepid} = c_2 - 1$ if swapped, or $c^g = c_2$ otherwise, so that it has the ordinal of the column-separating gap *physically* following the column c_2 .

The macro is used in `\pcol@ioutputelt`, `\pcol@addmarginpar`, `\pcol@imakeflushedpage` and `\pcol@iflushfloats` with $(c_1, c_2) = (c', c)$, and in `\pcol@addmarginpar` (another use) with $(c_1, c_2) = (c, c')$. Note that in the uses in the macros above except for `\pcol@addmarginpar`, `\c@page` definitely has the page number for the page to be shipped out. As for `\pcol@addmarginpar` on the other hand, `\c@page` can be different from the ship-out page number to produce a weird result if their parities are different, due to page jump. However this problem is not so severe because it just affects the position of marginal notes which L^AT_EX itself may misplace.

```
2663 \def\pcol@swapcolumn#1#2#3#4{%
2664   \edef\pcol@colsepid{\number#1}%
2665   \ifpcol@swapcolumn
2666     \ifodd\c@page\relax #2#1\relax
2667     \else
2668       #2#4\relax \advance#2-#1\relax \advance#2#3\relax \advance#2-\tw@
2669       \edef\pcol@colsepid{\number#2}%
2670       \advance#2\@ne
2671     \fi
2672   \else #2#1\relax
2673   \fi}
2674
```

`\marginparthreshold` The API macro `\marginparthreshold{ t_l }[t_r]` determines the smallest ordinal t_l of columns in left parallel-pages whose marginal notes go to the right margin in fundamental setting of `\pcol@marginparthreshold@l` marginal note positioning, while the threshold in right parallel-pages is given by t_r if provided or by t_l otherwise. That is, marginal notes given in a column c in a page p s.t. $c \in [0, C_L)$ (resp. $[C_L, C)$) go left if $c < t_l$ (resp. $c < t_r$) while they go right if $c \geq t_l$ (resp. $c \geq t_r$), providing

$$(\text{\ifpcol@swapmarginpar} \wedge \text{page}(p) \bmod 2 = 0) \neq \text{\if@reversemargin} = \text{false}$$

or the margins are swapped otherwise.

The macro defines `\pcol@mpthreshold@l` to let it have t_l after a assigning t_l to `\@tempcnta` to ensure t_l gives some number, and then do the same for `\pcol@mpthreshold@r` with t_r by `\pcol@marginparthreshold` if t_r is provided, or let the macro have t_l otherwise. Note that at the top level we do `\marginparthreshold{1}` to give defaults. Also note that `\pcol@mpthreshold@l` and `\pcol@mpthreshold@r` are referred to solely in `\pcol@addmarginpar`.

```
2675 \def\marginparthreshold#1{\@tempcnta#1\relax
2676   \xdef\pcol@mpthreshold@l{\number\@tempcnta}%
2677   \@ifnextchar[%]
2678     \pcol@marginparthreshold{\xdef\pcol@mpthreshold@r{\number\@tempcnta}}}
```

²³³Or the `\dimen` register `\z@`.

```

2679 \def\pcol@marginparthreshold[#1]{\@tempcnta#1\relax
2680 \xdef\pcol@mpthresholdr{\number\@tempcnta}}
2681 \marginparthreshold{1}
2682

```

21 Commands for Text Coloring

`\columncolor` The API macro `\columncolor[mode]{color}[c]` defines the default color specified by $\langle color \rangle$ optionally with color $\langle mode \rangle$ of the column c being the current column or that specified by the optional argument. After `\defining` `\pcol@colorcommand` being the `\string` of this macro itself, and processing two optional arguments $\langle mode \rangle$ and c through macros `\pcol@xcolumncolor`, `\pcol@ycolumncolor` and `\pcol@columncolor`, the macro `\pcol@icolumncolor{cmd}[c]` is invoked to perform real operations with the coloring command $\langle cmd \rangle = \color[mode]{color}$.

```

2683 %% Commands for Text Coloring
2684
2685 \def\columncolor{\def\pcol@colorcommand{\string\columncolor}%
2686 \ifnextchar[%]
2687 \pcol@xcolumncolor\pcol@ycolumncolor}
2688 \def\pcol@xcolumncolor[#1]#2{\pcol@columncolor{\color[#1]{#2}}}
2689 \def\pcol@ycolumncolor#1{\pcol@columncolor{\color{#1}}}
2690 \def\pcol@columncolor#1{\ifnextchar[%]
2691 {\pcol@icolumncolor{#1}}{\pcol@icolumncolor{#1}[\number\pcol@currcol]}}

```

`\normalcolumncolor` The API macro `\normalcolumncolor[c]` defines the default color of the column c , being the current column or that specified by the optional argument, is `\normalcolor`. That is, after `\defining` `\pcol@colorcommand` being the `\string` of this macro itself, this macro simply invokes `\pcol@icolumncolor{cmd}[c]` to perform real operations with the coloring command $\langle cmd \rangle = \normalcolor$.

```

2692 \def\normalcolumncolor{\def\pcol@colorcommand{\string\normalcolumncolor}%
2693 \ifnextchar[%]
2694 {\pcol@icolumncolor\normalcolor}%
2695 {\pcol@icolumncolor\normalcolor[\number\pcol@currcol]}}

```

`\pcol@icolumncolor` The macro `\pcol@icolumncolor{cmd}[c]`, invoked from `\pcol@columncolor` and `\normalcolumncolor`, performs the operations to define the default color of the column c with the coloring command $\langle cmd \rangle \in \{\color[mode]{color}, \normalcolor\}$ as follows. First we examine if `\set@color` is not `\relax` and we are in non-internal vertical or non-restricted horizontal mode and, if not, we complain the command whose name is in `\pcol@colorcommand` is ineffective by `\PackageWarning` and do nothing.

Otherwise and if we are not in a `paracol` environment, i.e., `\paracol` is not `\let`-equal to `\pcol@paracol`, we simply invoke `\pcol@iicolumncolor` to let $\hat{\gamma}_0^c = \pcol@columncolor \cdot c$ have the color χ specified by $\langle cmd \rangle$ so that the next `\begin{paracol}` will let γ_0^c have the coloring `\special` for χ . If we are in a `paracol` environment but in a column $c' \neq c$, on the other hand, we also let $\hat{\gamma}_0 = \chi$ but in addition let $\gamma_0^c = \pcol@columncolor@box \cdot c$ have the coloring `\special` for χ immediately so that it is effective in the next column-switching to c . This immediate setting of γ_0^c is done by invoking $\langle cmd \rangle$ with the original `\set@color` saved in `\pcol@set@color` and the nullification of `\aftergroup`, after acquiring an `\insert` for it if necessary,

Otherwise, i.e., if we are in a `paracol` environment and in the column c , at first we invoke `\pcol@scancst@shadow` to rewind $\hat{\Gamma}^c$ applying `\@elt = \reset@color` to $\hat{\gamma}_i \in \hat{\Gamma}^c$. Then, after

letting $\hat{\gamma}_0^c = \chi$, we invoke `\pcol@scancst@shadow` again to reestablish \hat{T}^c with the new $\hat{\gamma}_0^c$ so that $\hat{\gamma}_0^c$ is at the bottom of the color stack in `.tex`. In this scan `\@elt<\hat{\gamma}_i>` defines `\current@color` to let it have $\hat{\gamma}_i$ and then invokes `\pcol@set@color` to put the coloring `\special` for $\hat{\gamma}_i$ nullifying `\aftergroup`. Then we `\insert` a `\vbox`, whose height and depth are `1pt` and width is 0, having the coloring `\special` for χ so that `\output` will let γ_0^c have the `\special` in a synchronous manner. After that we put a `\penalty = 10000` if `\if@nobreak = true` to keep the `\insertion` from being followed by a page break.

The macro `\pcol@iicolumncolor<cmd><c>` at first invokes `<cmd>` to let `\current@color` have the printer-specific color information χ of `<color>` or what `\normalcolor` specifies, temporarily letting `\set@color` be `\relax` to let `\color` or `\normalcolor` just do the `\definition` of `\current@color` without putting coloring `\specials` nor preparing color stack popping. Then we `\xdefine` $\hat{\gamma}_0^c = \text{\pcol@columncolor}\cdot c$ to have χ .

The macro `\pcol@scancst@shadow` applies `\@elt` to $\hat{\gamma}_0^c$ to put a coloring or uncoloring `\special` for it if it is defined, and then do the same for all $\hat{\gamma}_i \in \hat{T} = \text{\pcol@colorstack@shadow}$.

```

2696 \def\pcol@iicolumncolor#1[#2]{%
2697   \@tempswafalse
2698   \ifpcol@inner \@tempwattrue \fi
2699   \ifinner      \@tempwattrue \fi
2700   \ifmode      \@tempwattrue \fi
2701   \ifx\set@color\relax
2702     \PackageWarning{paracol}{\pcol@colorcommand\space is not effective
2703       without some coloring package}%
2704   \else\if@tempswa
2705     \PackageWarning{paracol}{\pcol@colorcommand\space is not effective
2706       when not in outer par mode}%
2707   \else
2708     \begingroup
2709     \let\@elt\relax
2710     \ifx\pcol@paracol\paracol
2711       \pcol@iicolumncolor{#1}{#2}%
2712     \else\ifnum#2=\pcol@currcol
2713       \def\@elt##1{\reset@color}\pcol@scancst@shadow
2714       \pcol@iicolumncolor{#1}{#2}%
2715       \def\@elt##1{\def\current@color{##1}\let\aftergroup\@gobble
2716         \pcol@set@color}%
2717       \pcol@scancst@shadow
2718       \setbox\@tempboxa\vbox{\let\set@color\pcol@set@color
2719         \let\aftergroup\@gobble #1}%
2720       \ht\@tempboxa1sp \dp\@tempboxa1sp \wd\@tempboxa\z@\relax
2721       \insert\pcol@colorins{\box\@tempboxa}%
2722       \ifvmode\if@nobreak \nobreak \fi\fi
2723     \else
2724       \pcol@iicolumncolor{#1}{#2}%
2725       \pcol@currcol#2\relax
2726       \ifvoid\pcol@ccuse{@box}%
2727         \@next\@currbox\@freelist{}\pcol@ovf
2728         \pcol@ccxdef{\@currbox}%
2729       \fi
2730       \global\setbox\pcol@ccuse{@box}\vbox{\let\set@color\pcol@set@color
2731         \let\aftergroup\@gobble #1}%
2732     \fi\fi
2733   \endgroup
2734 \fi\fi

```

```

2735 \ignorespaces}
2736 \def\pcol@iicolumncolor#1#2{{\let\set@color\relax #1%
2737 \expandafter\xdef\csname pcol@columncolor#2\endcsname{\current@color}}}
2738 \def\pcol@scancst@shadow{%
2739 \pcol@ifccdefined{\@elt{\pcol@ccuse{}}}\relax
2740 \pcol@colorstack@shadow}
2741

```

`\pcol@mcpushlimit` The macro `\pcol@set@color@push` is invoked whenever L^AT_EX's counterpart `\set@color` appears in a `paracol` environment through coloring commands such as `\color`, because `\pcol@set@color` replaces L^AT_EX's macro with it saving the original version in `\pcol@set@color`, if the original `\set@color` is not `\relax` to mean some coloring package is in use. This original version is used through `\pcol@set@color` by `\pcol@bg@paintregion@i`, `\pcol@output@start` and `\pcol@iicolumncolor` besides this macro `\pcol@set@color@push`, while `\output@lets` `\set@color = \pcol@set@color` for the references outside of our control.

The macro at first invokes its original version being `\pcol@set@color` to put an appropriate coloring `\special` to `.dvi` and reserve the invocation of `\reset@color` by `\aftergroup`. Then, it performs one of two different operations depending on TeX's mode, i.e., math mode or not. If we are in math mode and not in a `\vbox`, at first we increment $m = \text{\pcol@mcid}$ and examine if $m > \text{\pcol@mcpushlimit} = 1000$, and if so we stop the execution with `\PackageError`²³⁴ in order to avoid too many macros `\pcol@reset@color@mpop@ m` are defined²³⁵. Otherwise, i.e., if $m \leq \text{\pcol@mcpushlimit}$, we reserve the invocation of the macro `\pcol@reset@color@mpop@ i` for our own pop by `\aftergroup` defining the macro as `\pcol@reset@color@mpop{ m }`. If we are not in math mode, on the other hand, and neither in a `\vbox` nor in restricted horizontal mode, we simply reserve the invocation of the macro .

Then, regardless that we are in math mode or not, we push the contents of `\current@color`, which `\set@color` should refer to as the color information to be set, into the shadow color stack $\hat{\Gamma} = \text{\pcol@colorstack@shadow}$ for the stack rewinding/reestablishing in `\columncolor` and `\normalcolumncolor`. Since this push is done non-`\globally` with `\edef`, we save/restore the definition of `\@elt` to/from `\pcol@elt@save` before/after the push, respectively²³⁶. Then we `\insert` a `\vbox` through `\pcol@colorins` for the push of γ_i or $\gamma_{i,m}$ to Γ_r synchronous with a page break or column-switching. The height of the `\vbox` is 1 pt, depth is 0 and width is m sp if we are in math mode or 0 otherwise, and its contents is the coloring `\special` given by `\pcol@set@color` so that the `\special` is what the macro put at the beginning of this macro but without the reservation of `\reset@color`. After the insertion, we put `\pcol@fcwhyphenate`, being `\hskip\z@` when `\coloredwordhyphenated` is effective, to split the coloring `\special` from the first colored word so that the word may be hyphenated if we are in horizontal mode. If we are in vertical mode, on the other hand, we do `\nobreak` if `\if@nobreak = true` to keep the `\insertion` from being followed by a page break.

```

2742 \def\pcol@mcpushlimit{1000}
2743 \def\pcol@set@color@push{\pcol@set@color
2744 \ifmmode\else\ifinner \pcol@innertrue \fi\fi
2745 \ifpcol@inner\else
2746 \ifmmode
2747 \global\advance\pcol@mcid\@ne

```

²³⁴And let $m = 1$ to allow a user to continue the execution bravely.

²³⁵We could make the number of math-mode coloring operations virtually unlimited by putting all digits of the decimal representation of m followed by a terminator by multiple `\aftergroups` so that `\pcol@reset@color@mpop` is put by `\aftergroup` prior to them to capture them as its argument, but limiting with $2^{31} - 1$ is still necessary and that with 1000 is reasonable.

²³⁶Just in case.

```

2748 \ifnum \pcol@mcid>\pcol@mcpushlimit\relax
2749 \PackageError{paracol}{Too many coloring commands in math mode}\@ehb
2750 \global\pcol@mdid\@ne
2751 \fi
2752 \@tempdima\pcol@mcid sp\relax
2753 \expandafter\aftergroup
2754 \csname pcol@reset@color@mpop@\number\pcol@mcid\endcsname
2755 \expandafter\xdef
2756 \csname pcol@reset@color@mpop@\number\pcol@mcid\endcsname
2757 {\noexpand\pcol@reset@color@mpop{\number\pcol@mcid}}%
2758 \else
2759 \aftergroup\pcol@reset@color@pop \@tempdima\z@
2760 \fi
2761 \let\pcol@elt@save\@elt \let\@elt\relax
2762 \edef\pcol@colorstack@shadow{\pcol@colorstack@shadow\@elt{\current@color}}%
2763 \let\@elt\pcol@elt@save
2764 \setbox\@tempboxa\vbox{\let\aftergroup\@gobble \pcol@set@color}%
2765 \ht\@tempboxa\z@ \dp\@tempboxa\z@ \wd\@tempboxa\@tempdima
2766 \insert\pcol@colorins{\box\@tempboxa}\ifhmode \pcol@fcwhyphenate \fi
2767 \ifvmode\if@nobreak \nobreak \fi\fi
2768 \fi}
2769

```

`\pcol@reset@color@pop` The macro `\pcol@reset@color@pop` and its math-mode relative `\pcol@reset@color@mpop` `{m}` are invoked by `\aftergroup` mechanism in `\pcol@set@color@push`, directly for the former and through the macro `\pcol@reset@color@mpop@m` for the latter. They `\insert` a `\vbox` for γ_i^- or $\gamma_{i,m}^-$ to add it to L_r synchronously with a page break or column-switching. Therefore, the height and depth of the `\vbox` are 0 and width is 0 for γ_i^- or m `sp` for $\gamma_{i,m}^-$. The contents of the `\vbox` is an uncoloring `\special` given by `\reset@color` but this is done just for debugging to show what `\pcol@colorins` has by, for example, `\pcol@ShowBox`. Then if we are in vertical mode and `\if@nobreak = true`, we do `\nobreak` to keep the `\insertion` from being followed by a page break even in `\pcol@reset@color@mpop` because its corresponding `\pcol@set@color@push` may have been in a displayed math construct after which we are in vertical mode.

One caution is that `\pcol@reset@color@pop` can be invoked outside the `paracol` environment in which the corresponding `\pcol@set@color@push` appears. In this case with `\ifpcol@output = false`, we don't need to do the pop operation and cannot make the `\insertion` for it because `\output` is not for `paracol`.

```

2770 \def\pcol@reset@color@pop{%
2771 \ifpcol@output
2772 \setbox\@tempboxa\vbox{\reset@color}%
2773 \ht\@tempboxa\z@ \dp\@tempboxa\z@ \wd\@tempboxa\z@
2774 \insert\pcol@colorins{\box\@tempboxa}%
2775 \ifvmode\if@nobreak \nobreak \fi\fi
2776 \fi}
2777 \def\pcol@reset@color@mpop#1{%
2778 \setbox\@tempboxa\vbox{\reset@color}%
2779 \ht\@tempboxa\z@ \dp\@tempboxa\z@ \wd\@tempboxa#1sp\relax
2780 \insert\pcol@colorins{\box\@tempboxa}%
2781 \ifvmode\if@nobreak \nobreak \fi\fi
2782 }
2783

```

`\coloredwordhyphenated` The API macro `\coloredwordhyphenated` defines the macro `\pcol@fcwhyphenate` being `\hskip\z@` so that the null space is inserted after the coloring `\special` and `\insert` put by `\pcol@set@color@push` if we are in horizontal mode, so that the word following them can be hyphenated. The other API macro `\nocoloredwordhyphenated` makes `\pcol@fcwhyphenate = \relax` to inhibit the insertion. Since the null skip is a line break candidate, the skip might cause an unexpected and undesirable line break. However, this demerit is less important than the merit of making it possible to hyphenate the first word in multi-column documents with narrow lines, and thus we make `\coloredwordhyphenated` effective in default while giving users a means to disable the insertion (occasionally) by `\nocoloredwordhyphenated`.

```

2784 \def\coloredwordhyphenated{\def\pcol@fcwhyphenate{\hskip\z@}}
2785 \def\nocoloredwordhyphenated{\let\pcol@fcwhyphenate\relax}
2786 \coloredwordhyphenated
2787

```

22 Commands for Column-Separating Rule Color and Background Painting

`\colseprulecolor` The macro `\colseprulecolor[mode]{color}[c]` defines `\pcol@colseprulecolor` to have the `<color>` optionally with coloring `<mode>` of all column-separating rules if the optional argument `c` is not provided, or `\pcol@colseprulecolor·c` that of the rule drawn between a particular column pair `c` and `c+1`. After defining `\pcol@colorcommand` to be `\colseprulecolor` in case we have to give a warning, the macro invokes `\pcol@defcseprulecolor@x[mode]{color}` or `\pcol@defcseprulecolor@y{color}` according to the provision of the optional argument `<mode>` to invoke `\pcol@defcseprulecolor` with argument `{cmd} = \color[mode]{color}` so that this macro invokes `\pcol@defcseprulecolor@i{cmd}[c]` where `c = ∅` if the optional argument `c` is not provided.

The macro `\normalcolseprulecolor[c]`, on the other hand, defines `\pcol@colseprulecolor[c]` with whatever `\normalcolor` gives, and thus it invokes `\pcol@defcseprulecolor@i` letting `<cmd> = \normalcolor`, after defining `\pcol@colorcommand` to be `\normalcolseprulecolor`.

The macro `\pcol@defcseprulecolor@i{cmd}[c]` examines if `\set@color = \relax` to mean no coloring packages have been loaded and, if so, do nothing giving a warning the command in `\pcol@colorcommand` is not effective. Otherwise, we examine if `<cmd>` has proper arguments by invoking it but temporally nullifying `\set@color` and then define `\pcol@colseprulecolor[·c]` to be `<cmd>`.

Note that at the top level we define `\pcol@colseprulecolor` to be `\normalcolor` to give the default for all column-separating rules. Also note that macros `\pcol@colseprulecolor·c` are referred to solely in `\pcol@hfil` which also uses `\pcol@colseprulecolor` for columns for which `\pcol@colseprulecolor·c` is not defined.

```

2788 %% Commands for Column-Separating Rule Color and Background Painting
2789
2790 \def\colseprulecolor{\def\pcol@colorcommand{\string\colseprulecolor}%
2791 \@ifnextchar[%]
2792   \pcol@defcseprulecolor@x\pcol@defcseprulecolor@y}
2793 \def\pcol@defcseprulecolor@x[#1]#2{\pcol@defcseprulecolor{\color[#1]{#2}}}
2794 \def\pcol@defcseprulecolor@y#1{\pcol@defcseprulecolor{\color{#1}}}
2795 \def\pcol@defcseprulecolor#1{\@ifnextchar[%]
2796   {\pcol@defcseprulecolor@i{#1}}{\pcol@defcseprulecolor@i{#1}[]}}
2797 \def\normalcolseprulecolor{%

```



```

2798 \def\pcol@colorcommand{\string\normalcolseprulecolor}%
2799 \@ifnextchar[%]
2800   {\pcol@defcseprulecolor@i\normalcolor}%
2801   {\pcol@defcseprulecolor@i\normalcolor[]}}
2802 \def\pcol@defcseprulecolor@i#1[#2]{%
2803   \ifx\set@color\relax
2804     \PackageWarning{paracol}{\pcol@colorcommand\space is not effective
2805       without some coloring package}%
2806   \else
2807     {\let\set@color\relax #1}%
2808     \global\@namedef{pcol@colseprulecolor#2}{#1}%
2809   \fi}
2810 \gdef\pcol@colseprulecolor{\normalcolor}
2811

```

`\backgroundcolor` The macro `\backgroundcolor{region}[mode]{color}` defines the $\langle color \rangle$ optionally with $\langle mode \rangle$ of the $\langle region \rangle$ whose syntax is specified as follows.

```

\pcol@backgroundcolor@e
\pcol@backgroundcolor
\pcol@backgroundcolor@i
\pcol@bg@region
\pcol@backgroundcolor@ii
\pcol@backgroundcolor@iii
\pcol@backgroundcolor@iv
\pcol@backgroundcolor@v

```

$$\langle region \rangle ::= \langle regionid \rangle \langle extension \rangle$$

$$\langle regionid \rangle ::= \langle a \rangle | \langle corg \rangle [c]$$

$$\langle a \rangle ::= \langle corg \rangle | s | S | t | T | l | L | r | R | f | F | n | N | p | P$$

$$\langle corg \rangle ::= c | C | g | G$$

$$\langle extension \rangle ::= \emptyset | (x_0, y_0) | (x_0, y_0) (x_1, y_1)$$

On the other hand, the counterpart macro `\nbackgroundcolor{region}` undefines the color of $\langle region \rangle$. Both macros invoke `\pcol@backgroundcolor` giving all arguments to it to parse the argument $\langle region \rangle$, after letting $f_{def} = \text{\if@tempswa = true}$ and `\pcol@backgroundcolor@e` = `\pcol@backgroundcolor@w` in `\backgroundcolor`, or $f_{def} = \text{\if@tempswa = true}$ and `\pcol@backgroundcolor@e` = `\pcol@backgroundcolor@z` in `\nbackgroundcolor`. Note that the `\let`-assignment to `\pcol@backgroundcolor@e` is effective when we found an error in the parse of the argument $\langle region \rangle$ to throw away what remains in the argument unprocessed and, for `\backgroundcolor`, the other arguments $[mode]\{color\}$.

Then the macro `\pcol@backgroundcolor` examines if `\pcol@bg@@a` is defined, and if not, raises an error that a is invalid and then, in case the user dare to continue the execution, invokes `\pcol@backgroundcolor@e` to throw all arguments away after letting $a' = \text{\pcol@bg@region} = xx$ so that the undefining `\pcol@bg@color·a'` does not cause any troubles. Otherwise, i.e., if `\pcol@bg@@a` is defined, it invokes `\pcol@backgroundcolor@i[c]` or `\pcol@backgroundcolor@ii` according to the provision of the optional argument $[c]$, after `\defining` $a' = \text{\pcol@bg@region}$ to be a . Then `\pcol@backgroundcolor@i` examines if `\pcol@bg@may` have `col@a` is defined, and if not, raises an error again in a similar way, or otherwise invokes `\pcol@backgroundcolor@ii` after re`\defining` $a' = a·c$.

Then if $f_{def} = \text{true}$, the macro `\pcol@backgroundcolor@ii` examines if `\set@color = \relax`, and if so it complains that any coloring packages have not been loaded and invokes `\pcol@backgroundcolor@w` just for throwing away optional arguments for extension and $[mode]\{color\}$. Otherwise, i.e., if `\set@color ≠ \relax`, it invokes `\pcol@backgroundcolor@iii` after adding a' to the tail of `\pcol@bg@defined`. On the other hand if $f_{def} = \text{false}$, it invokes `\pcol@backgroundcolor@z` without checking the availability of coloring macros.

The macro `\pcol@backgroundcolor@iii` at first invokes `\pcol@bg@defext{d}{e}` with $e = 0$ for all $d \in \{1, r, t, b\}$ to have default (no) extensions. Then if (x_0, y_0) is provided, `\pcol@bg@defext` is invoked again by `\pcol@backgroundcolor@iv` for all $d \in \{1, r, t, b\}$ but with $e = x_0$ for $d \in \{1, r\}$ and with $e = y_0$ for $d \in \{t, b\}$. Further, if (x_1, y_1) is also provided, `\pcol@`

`bg@defext` is invoked once again by `\pcol@backgroundcolor@v` for all $(d, e) \in \{(r, x_1), (b, y_1)\}$. Finally `\pcol@backgroundcolor@v` invokes `\pcol@backgroundcolor@x`, which is also invoked from `\pcol@backgroundcolor@iii` and `\pcol@backgroundcolor@iv` if they find no (further) extensions.

```

2812 \def\backgroundcolor#1{\@tempswattrue
2813 \let\pcol@backgroundcolor@e\pcol@backgroundcolor@w
2814 \pcol@backgroundcolor#1\@nil}
2815 \def\nobackgroundcolor#1{\@tempswafalse
2816 \let\pcol@backgroundcolor@e\pcol@backgroundcolor@z
2817 \pcol@backgroundcolor#1\@nil}
2818 \def\pcol@backgroundcolor#1{%
2819 \@ifundefined{pcol@bg@#1}%
2820 {\PackageError{paracol}%
2821 {Invalid background coloring region identifier #1}%
2822 \def\pcol@bg@region{xx}\pcol@backgroundcolor@e}%
2823 {\def\pcol@bg@region{#1}%
2824 \@ifnextchar [%]
2825 \pcol@backgroundcolor@i \pcol@backgroundcolor@ii}}
2826 \def\pcol@backgroundcolor@i[#1]{%
2827 \@ifundefined{pcol@bg@mayhavecol@pcol@bg@region}%
2828 {\PackageError{paracol}%
2829 {Column number \number#1 is not effective for background coloring region
2830 \pcol@bg@region}%
2831 \def\pcol@bg@region{xx}\pcol@backgroundcolor@e}%
2832 {\edef\pcol@bg@region{\pcol@bg@region @#1}%
2833 \pcol@backgroundcolor@ii}}
2834 \def\pcol@backgroundcolor@ii{%
2835 \if@tempswa
2836 \ifx\set@color\relax
2837 \PackageWarning{paracol}{\string\backgroundcolor\space is not effective
2838 without some coloring package}%
2839 \let\reserved@b\pcol@backgroundcolor@w
2840 \else
2841 \let\reserved@b\pcol@backgroundcolor@iii
2842 \@cons\pcol@bg@defined{\pcol@bg@region}}%
2843 \fi
2844 \else
2845 \let\reserved@b\pcol@backgroundcolor@z
2846 \fi
2847 \reserved@b}
2848 \def\pcol@backgroundcolor@iii{%
2849 \pcol@bg@defext{l}\z@ \pcol@bg@defext{r}\z@
2850 \pcol@bg@defext{t}\z@ \pcol@bg@defext{b}\z@
2851 \@ifnextchar (%)
2852 \pcol@backgroundcolor@iv \pcol@backgroundcolor@x}
2853 \def\pcol@backgroundcolor@iv(#1,#2){%
2854 \pcol@bg@defext{l}-{#1}\pcol@bg@defext{r}-{#1}%
2855 \pcol@bg@defext{t}-{#2}\pcol@bg@defext{b}-{#2}%
2856 \@ifnextchar (%)
2857 \pcol@backgroundcolor@v \pcol@backgroundcolor@x}
2858 \def\pcol@backgroundcolor@v(#1,#2){%
2859 \pcol@bg@defext{r}-{#1}\pcol@bg@defext{b}-{#2}%
2860 \pcol@backgroundcolor@x}

```

`\pcol@backgroundcolor@x` The macro `\pcol@backgroundcolor@x` is used in `\pcol@backgroundcolor@iii`, `\pcol@backgroundcolor@iv` and `\pcol@backgroundcolor@v` to define the color for background painting of the region $a' = \text{\pcol@bg@region}$. Since the macro is followed by the arguments $[mode]\{color\}$ of `\backgroundcolor`, the macro invokes `\color` to let it `\define\current@color` but without real coloring operations by letting `\set@color = \pcol@backgroundcolor@y`. Therefore `\pcol@backgroundcolor@y` is invoked in `\color` and it `\xdefines\pcol@bg@colpr@a'` to let it have whatever `\current@color` has.

```

2861 \def\pcol@backgroundcolor@x#1\@nil{\begingroup
2862   \let\set@color\pcol@backgroundcolor@y \color}
2863 \def\pcol@backgroundcolor@y{%
2864   \expandafter\xdef\csname pcol@bg@color@\pcol@bg@region\endcsname
2865     {\current@color}%
2866   \endgroup}

```

`\pcol@backgroundcolor@z` The macro `\pcol@backgroundcolor@z` is invoked from `\pcol@backgroundcolor@ii` directly `\pcol@backgroundcolor@w` to work for `\nobackgroundcolor` to disable the background painting for a region $a' = \text{\pcol@bg@region}$, or from `\pcol@backgroundcolor` and `\pcol@backgroundcolor@i` through `\pcol@bg@color@xx` `\pcol@backgroundcolor@e` when they find an error in the argument $\langle region \rangle$ of `\nobackgroundcolor`. Similarly the macro `\pcol@backgroundcolor@w` is invoked from `\pcol@backgroundcolor@ii` directly when it finds no coloring packages have not been loaded, or `\pcol@backgroundcolor` and `\pcol@backgroundcolor@i` through `\pcol@backgroundcolor@e` on error too but in the argument of `\backgroundcolor`. Both macros throw away whatever remains in $\langle region \rangle$ unprocessed and then invoke `\pcol@backgroundcolor@wi`, but `\pcol@backgroundcolor@z` gives it a dummy argument pair, while `\pcol@backgroundcolor@w` passes $[mode]\{color\}$ to it.

Then `\pcol@backgroundcolor@wi` throw all arguments away and lets `\pcol@bg@color@a' = \relax` so that the region a' is untouched in background painting macros. Note that since $a' = xx$ being an absolutely non-existent region when this macro is used for error recovery, undefining `\pcol@bg@color@xx` is not harmful.

```

2867 \def\pcol@backgroundcolor@z#1\@nil{\pcol@backgroundcolor@wi []{}}
2868 \def\pcol@backgroundcolor@w#1\@nil{\@ifnextchar[%]
2869   \pcol@backgroundcolor@wi{\pcol@backgroundcolor@wi []}}
2870 \def\pcol@backgroundcolor@wi[#1]#2{%
2871   \expandafter\global\expandafter\let
2872     \csname pcol@bg@color@\pcol@bg@region\endcsname\relax}
2873

```

`\pcol@bg@mayhavecol@c` The macros `\pcol@bg@mayhavecol@a` where $a \in \{c, C, g, G\}$ are used in `\pcol@backgroundcolor@i` when the region specifier a in $\langle region \rangle$ argument of `\backgroundcolor` or `\nobackgroundcolor` is followed by optional $[c]$, so that the invoker macro examines if a can have the optional column ordinal. Therefore, the macros just need not to be `\relax` and thus commonly have empty bodies.

```

2874 \def\pcol@bg@mayhavecol@c{}
2875 \def\pcol@bg@mayhavecol@C{}
2876 \def\pcol@bg@mayhavecol@g{}
2877 \def\pcol@bg@mayhavecol@G{}
2878

```

`\pcol@bg@defext` The macro `\pcol@bg@defext\{d\}\{e\}` is used by `\pcol@backgroundcolor@iii`, `\pcol@backgroundcolor@iv` and `\pcol@backgroundcolor@v` to `\define\pcol@bg@ext@d@a'` be $e = 0$ for all $d \in \{1, r, t, b\}$ in the first, be $e = x_0$ for $d \in \{1, r\}$ and $e = y_0$ for $d \in \{t, b\}$ in the

second, and be $e = x_0, x_1, y_0$ and y_1 for d being **l**, **r**, **t**, **b** respectively in the last. The macro at first lets `\@tempdima` have e to confirm e is a proper dimension and then `\xdefines` `\pcol@bg@ext@d@a'` to let it have the integer representation of e followed by `sp`.

```
2879 \def\pcol@bg@defext#1#2{%
2880   \@tempdima#2\relax
2881   \expandafter\xdef\csname pcol@bg@ext@#1\pcol@bg@region\endcsname{%
2882     \number\@tempdima sp}}
2883
```

`\resetbackgroundcolor` The API macro `\resetbackgroundcolor` disables background painting of all regions whose colors have been specified by `\backgroundcolor`. Since the region specifiers a'_1, a'_2, \dots, a'_n for which background painting is specified are recorded in `\pcol@bg@defined = \@elt{a'_1}\@elt{a'_2}\dots\@elt{a'_n}` by `\pcol@backgroundcolor`, the macro invokes `\pcol@bg@defined` temporarily letting `\@elt = \pcol@resetbackgroundcolor` to let `\pcol@bg@color@a'_i` be `\relax` for all $i \in [1, n]$, and clears `\pcol@bg@defined`, whose initial state is also empty.

```
2884 \def\resetbackgroundcolor{%
2885   \let\@elt\pcol@resetbackgroundcolor \pcol@bg@defined
2886   \gdef\pcol@bg@defined{}}
2887 \def\pcol@resetbackgroundcolor#1{%
2888   \expandafter\global\expandafter\let\csname pcol@bg@color@#1\endcsname\relax}
2889 \gdef\pcol@bg@defined{}
2890
```

23 Closing Environment

`\endparacol` The macro `\endparacol` is invoked from `\end{paracol}` to close `paracol` environment. After `\pcol@lastcol` making it sure to be in vertical mode by `\pcol@par`, we switch to the column 0 by `\pcol@switchcol` to let local counter have the values for the column 0 so that they are referred to outside the environment, after saving the current column c in `\pcol@lastcol` to be referred to in `\pcol@output@end` so that $\kappa_c(\sigma)$ and $\kappa_c(\varepsilon)$ are passed to post-environment stuff. In `\pcol@switchcol`, we make sure, not to add the preamble for column 0 by checking whether `\pcol@lastcol` is set.

Then we invoke `\pcol@flushclear` for pre-flushing column height check, turning `\ifpcol@lastpage = true` to tell `\pcol@output@switch` for the check that it works on the last page, and giving \perp to `\pcol@flushclear` as its argument, unless footnote typesetting is page-wise but not merged for which we give Φ to ensure all deferred footnotes are put in the checking process. Note that the argument for column-wise footnote typesetting is Φ but it is definitely \perp in this mode. After that we make an `\output` request by `\pcol@invokeoutput` with `\penalty = \pcol@op@end` and still with `\ifpcol@lastpage = true` to build the last page.

Next, we let `\columnwidth = \textwidth` and `\if@twocolumn = false` for single-column typesetting, and also let `\topskip = \pcol@topskip` to make it sure that the parameter has the value used outside in `paracol` environment. Finally, if the footnote counter adjustment is required by `\ifpcol@fncounteradjustment = true`, we let `\c@footnote = b_f + n_f`.

```
2891 %% Closing Environment
2892
2893 \def\endparacol{\pcol@par
2894   \edef\pcol@lastcol{\number\pcol@currcol}}
2895 \pcol@nextcol\z@ \pcol@switchcol
```

```

2896 \pcol@lastpagetrue
2897 \ifpcol@mgfnote \pcol@flushclear\voidb@x
2898 \else \pcol@flushclear\pcol@topfnnotes
2899 \fi
2900 \pcol@invokeoutput\pcol@op@end
2901 \global\columnwidth\textwidth
2902 \global@twocolumnfalse
2903 \global\topskip\pcol@topskip
2904 \ifpcol@fncounteradjustment
2905   \global\c@footnote\pcol@footnotebase
2906   \global\advance\c@footnote\pcol@nfootnotes
2907 \fi}
2908

```

`\pcol@restoreeveryvbox` The macro `\pcol@restoreeveryvbox` is invoked just after `\end{paracol}` by `\aftergroup` mechanism activated by `\pcol@zparacol`. It examines if `\pcol@everyvbox` has tokens different from `\pcol@dummysymbol` which `\pcol@zparacol` `\globally` assigned to the register. Since the dummy token cannot be assigned to `\everyvbox`²³⁷, the difference means the `\everyvbox` has been `\globally` updated with the value that `\pcol@everyvbox` has now. Therefore if so, we globally update `\everyvbox` with `\pcol@everyvbox` to reflect the global update in the environment.

```

2909 \def\pcol@restoreeveryvbox{%
2910   \expandafter\def\expandafter\reserved@a\expandafter{\the\pcol@everyvbox}%
2911   \def\reserved@b{\pcol@dummysymbol}%
2912   \ifx\reserved@a\reserved@b\else \global\everyvbox\pcol@everyvbox \fi}

```

²³⁷Unless a very surprising coincidence happens or a user intentionally violates the coherence of the implementation.

Acknowledgments

The author thanks to Yacine Daddi Addoun who gave the author the motivation to write the style for his bilingual document. He also thanks to the following people; Robin Fairbairns who kindly invited the style to CTAN after the author's lazy six years failing to upload the style; Joseph G. Rosenstein and Dieter Köhler who suggested the author adding the function of unbalanced column width incorporated in version 1.1; Joaquín Blas who motivated the author to challenge page-wise footnotes; Olivier Vogel who pointed out the compatibility problem with coloring packages; Heiner Richter who asked for the possibility of swapping unbalanced columns, revealed two bugs in version 1.22 related to coloring and float pages, showed the necessity of `\coloredwordhyphenated`, and finally found the necessity of `\globalcounter*`; an anonymous user who reported a very rare-case but severe bug in the version 1.1 by which a page can be lost (whoops!); Olivier Gerard who found another terrible bug fixed in version 1.21 but hidden in `paracol` for two years by which a column disappears or moves to a wrong page (another whoops!), suggested to implement `\setcolumnwidth`, `\marginparthreshold` and `\thecolumn` introduced in version 1.3, and kindly proofread this manual; George Kamel who let the author know the coloring function newborn in version 1.2 had a bug fixed in version 1.22 to which he also made a great contribution testing many tentative versions with his own colored documents; another anonymous user who pointed out version 1.22 had yet another coloring bug fixed in version 1.24; Jean Druel who motivated the author to implement an advanced functionality parallel-paging; Tilo Arens and other patient users who had wished `paracol` would have the capability of rule drawing in the gaps separating columns and painting backgrounds of columns and so on; Michael Bolin who gave the author motivated examples showing the necessity of `\ensurevspace`. Tigran Aivazian who reported a memory leak problem fixed in version 1.32; Marcus Zelezny and Touhami Mamouni who found an incompatibility with L^AT_EX itself (2015/01/10 or later) and enlighten the author on the cause of the problem; Manuel Kuehner who reported a bug in text coloring which had hidden for five years until the version 1.34 was released; ZongXian Wang who found that the `paracol` misbehaves when an environment starts with an unusually tall item; and Frank Mittelbach who pointed out bugs in `\marginpar` implementation and vertical spacing with `\trivlist`-like environments, and suggested new functionality with `\marginnote`, `\belowfootnoteskip` and `\definecolumnpreamble`.

For the implementation of the style file, the author referred to the base implementations of `\output` and othe many macros of L^AT_EX 2_ε written by Leslie Lamport, Johannes Braams and other authors. The author also referred to `color` written by David Carlisle and `marginnote` written by Markus Kohm to make the package working well with them.

Index

Underlined number refers to the page where the implementation or the definition of the corresponding entry is described, while italicized number is for the page in which the specification or usage of the entry is explained. To find a control sequence, remove prefixes `\@`, `\if@`, `\pcol@` and `\ifpcol@` from its name if it has one of them.

Symbols	
Γ	<i>70, 71, 72, 100, 101, 104, 139, 140, 188, 189, 191, 225</i>
Γ_r	<i>71, 71, 73, 74, 88, 101, 104, 119, 127, 139, 140, 188, 189, 262, 263</i>
Γ^c	<i>71, 72–74, 100, 104, 114, 139, 178, 180, 187, 189, 191</i>
Γ_r^c	<i>71, 139, 188, 189</i>
$\hat{\Gamma}$	<i>72, 120, 230, 261, 262</i>
$\hat{\Gamma}^c$	<i>73, 73, 114, 120, 260, 261</i>
Γ_s	<i>74, 74, 100, 101, 139, 140, 148, 164, 178, 180, 187–189, 191</i>
Θ	<i>68, 68, 120, 126–128, 228, 242–244</i>
Θ^g	<i>68, 68, 105, 113, 120, 121, 126, 127, 228, 242</i>
Θ^l	<i>68, 68, 113, 120, 228, 229, 242–245</i>
Θ_c	<i>68, 68, 85, 105, 113, 120, 121, 228, 229, 243–245, 248</i>
Π	<i>66, 67, 68, 112, 118, 121, 125, 126, 130, 132, 149–152, 154, 155, 175, 221</i>
Π^+	<i>68, 94, 118, 119, 125, 130, 149, 151, 152, 195, 198</i>
Φ	<i>69, 69, 82, 88, 92, 96, 98, 101, 133, 134, 138, 143, 164, 175, 183, 192, 193, 225, 253–255, 268</i>
γ_i	<i>70, 70, 71, 73, 96, 101, 104, 114, 125, 188–190, 262</i>
$\gamma_{i,m}$	<i>73, 73, 74, 85, 96, 101, 114, 119, 127, 188, 189, 262</i>
γ_0^c	<i>71, 71–73, 99–101, 104, 110, 112–114, 121, 122, 127, 128, 139, 178, 188, 189, 191, 225, 260, 261</i>
γ_i^-	<i>71, 71, 73, 101, 188, 189, 263</i>
$\gamma_{i,m}^-$	<i>73, 73, 74, 101, 119, 127, 188, 189, 263</i>
$\hat{\gamma}_i$	<i>72, 72, 73, 120, 260, 261</i>
$\hat{\gamma}_0^c$	<i>72, 73, 99, 110, 114, 120, 128, 178, 188, 191, 260, 261</i>
$\zeta(\theta)$	<i>68, 68, 120, 124, 228, 242, 245</i>
θ^g	<i>68, 68, 115, 127, 228, 242, 245</i>
θ^l	<i>68, 68, 228, 229, 242–245, 248</i>
κ_c	<i>65, 65, 81, 83, 84, 86, 88, 89, 92–94, 96, 99, 100, 102–105, 111–113, 117–124, 126, 130, 132, 135–138, 145, 147, 148, 177, 178, 180–187, 192, 201, 202, 204–215, 219–222, 225–227, 249, 250, 268</i>
$\kappa_c(\beta)$	<i>65, 65, 81, 92, 93, 96, 100, 111–113, 119, 121, 122, 124, 130, 132, 135, 137, 138, 145, 177, 178, 180, 181, 183–185, 187, 201, 204–206, 208, 209, 211–213, 215, 217, 219, 222, 225</i>
$\kappa_c(\delta)$	<i>65, 88, 137, 177, 185, 202, 209–213, 223–227</i>
$\kappa_c(\varepsilon)$	<i>66, 102, 178, 181, 182, 185, 225, 249, 250, 268</i>
$\kappa_c(\eta)$	<i>65, 94, 185</i>
$\kappa_c(\lambda_b)$	<i>65, 119, 120, 124, 177, 185, 201, 205, 206, 209, 210</i>
$\kappa_c(\lambda_d)$	<i>65, 103–105, 113, 117–121, 123, 126, 136, 177, 185, 206–208, 210, 220, 221</i>
$\kappa_c(\lambda_m)$	<i>65, 119, 124, 177, 185, 205</i>
$\kappa_c(\lambda_t)$	<i>65, 93, 104, 105, 113, 118–122, 124, 145, 177, 185, 201, 205–207, 209, 210, 219</i>
$\kappa_c(\nu_b)$	<i>65, 84, 177, 185</i>
$\kappa_c(\nu_c)$	<i>65, 84, 185</i>
$\kappa_c(\nu_t)$	<i>65, 84, 177, 185, 214</i>
$\kappa_c(\xi)$	<i>65, 96, 137, 145, 147, 148, 185, 201, 209, 211–213</i>
$\kappa_c(\rho_b)$	<i>65, 93, 178, 185</i>
$\kappa_c(\rho_t)$	<i>65, 89, 93, 105, 177, 185, 205, 207, 208, 219</i>
$\kappa_c(\sigma)$	<i>65, 65, 83, 86, 103, 104, 178, 181, 182, 185, 226, 249, 268</i>
$\kappa_c(\tau)$	<i>65, 65, 81, 99, 113, 122, 177, 181, 185, 186, 192, 201, 202, 204, 205, 209, 210, 219</i>
μ	<i>89, 90, 137, 137, 185, 227, 229, 247</i>
$\pi(p)$	<i>66, 66–70, 76, 77, 80, 81, 84–86, 88, 89, 91–97, 99, 100, 104, 105, 107, 112, 113, 118–123, 125, 131–133, 138, 143, 145, 146, 148–159, 164, 175, 176, 180, 181, 183, 184, 186, 192, 194, 195, 197–200, 203–208, 211, 215–219, 225</i>

A

<code>\pcol@ac@caption</code>	<i>105, 106, 110, 115, 121, 251, 252, 252</i>
<code>\pcol@ac@caption@def</code>	<i>105, 115, 251, 251, 252</i>
<code>\pcol@ac@caption@disable</code>	<i>105, 110, 251, 251</i>
<code>\pcol@ac@caption@enable</code>	<i>105, 110, 251, 251</i>
<code>\pcol@ac@caption@if@lof</code>	<i>105, 110, 251</i>

<code>\pcol@ac@caption@if@lot</code>	... 105, 110, 251	<code>\pcol@backgroundcolor@e</code>	... 265 , 265, 267		
<code>\pcol@ac@caption@latex</code>	... 115, 252 , 252	<code>\pcol@backgroundcolor@i</code>		
<code>\pcol@ac@def@t</code> 110, 251	105, 108, 110, 265 , 265, 267		
<code>\pcol@ac@def@lof</code> 110, 251 , 251	<code>\pcol@backgroundcolor@ii</code>		
<code>\pcol@ac@def@lot</code> 110, 251 , 251	105, 108, 113, 114, 127, 265 , 265, 267		
<code>\pcol@ac@def@toc</code> 115, 251 , 251	<code>\pcol@backgroundcolor@iii</code>		
<code>\pcol@ac@disable@toc</code>	.. 106, 115, 251 , 251	91, 111, 265 , 265–267		
<code>\pcol@ac@enable@toc</code> 115, 251 , 251	<code>\pcol@backgroundcolor@iv</code>	111, 265 , 265–267		
<code>\pcol@aonly</code> 248, 250 , 251	<code>\pcol@backgroundcolor@v</code>	... 265 , 266, 267		
<code>\pcol@aonlyelt</code>	110, 120, 128, 248, 251 , 251	<code>\pcol@backgroundcolor@w</code>	110, 265, 267 , 267		
<code>\addcontentsline</code>	105, 106 , 115, 248, 251, 252	<code>\pcol@backgroundcolor@wi</code> 267 , 267		
<code>\pcol@addcontentsline</code>	.. 106, 251 , 251, 252	<code>\pcol@backgroundcolor@x</code>		
<code>\addcontentsonly</code>	107, 114, 123, 127, 266 , 266, 267		
.....	29, 29, 108, 110, 113, 248, 250 , 250	<code>\pcol@backgroundcolor@y</code>	114, 123, 266 , 267		
<code>\pcol@addflhd</code>	84, 88, 92, 94, 98, 104, 119,	<code>\pcol@backgroundcolor@z</code>		
.....	120, 124, 137, 145, 180, 209, 210 , 210	123, 127, 265, 267 , 267		
<code>\@addmarginpar</code>	base page 64 , 130		
.....	75, 94, 118 , 118, 194 , 194, 195, 229	<code>\baselineskip</code>		
<code>\pcol@addmarginpar</code>	19, 97, 105, 106, 132, 144, 163,		
.....	67, 75, 83, 85–87, 90, 91, 93,	166, 175–177, 187, 203, 213, 248, 250		
.....	95, 96, 102, 103, 111, 112, 118, 119,	<code>\pcol@basepage</code> 64 , 130 , 130		
.....	121, 122, 125, 127–130, 135, 136,	<code>\begin</code>	6–8, 10, 15–20, 25, 26, 30, 32, 41,		
.....	150, 194 , 194, 197, 198, 229, 257, 259	44, 55, 57, 60, 66, 68, 72, 74, 87, 92,		
<code>\pcol@@addmarginpar</code>	94, 123, 135, 138, 143, 174, 175, 177,		
.....	94, 102, 118, 194 , 194–196, 229	178, 188, 224, 227–229, 232, 249, 260		
<code>\addpenalty</code> 106 , 183	<code>\begingroup</code> 73		
<code>\@addtobot</code> 88, 92, 143, 147	<code>\@beginparpenalty</code> 84 , 106, 228		
<code>\addtocontents</code> 106	<code>\belowfootnoteskip</code>		
<code>\addtocounter</code> 31	15, 25, 91, 138 , 138, 175, 193		
<code>\addtodblcol</code> 80, 123, 153	<code>\ifpcol@bfbottom</code>	.. 108, 136 , 163, 201, 210		
<code>\addvspace</code> 106	<code>\pcol@bg@@a</code> 110, 168, 170, 172 , 265		
<code>\pcol@adjustfnctr</code> 110, 256 , 256	<code>\pcol@bg@@B</code> 172		
<code>\advance</code> 169	<code>\pcol@bg@@b</code> 169, 172		
<code>\afterassignment</code> 241	<code>\pcol@bg@@C</code> 172		
<code>\aftergroup</code> 70,	<code>\pcol@bg@@c</code> 172		
.....	72, 73, 110, 134, 178, 230, 260–263, 269	<code>\pcol@bg@@F</code> 172		
<code>\if@afterindent</code> 65, 83, 86, 103 ,	<code>\pcol@bg@@f</code> 90, 172		
.....	103, 127, 132, 178, 181, 182, 186, 226	<code>\pcol@bg@@G</code> 172		
B				<code>\pcol@bg@@g</code> 172
B_a 76, 76, 78, 154, 156, 168	<code>\pcol@bg@@L</code> 172		
b_f 85, 130 , 130, 134, 231, 256, 268	<code>\pcol@bg@@l</code> 169, 172		
background painting 21,	<code>\pcol@bg@@N</code> 172		
.....	27, 27–29, 48, 49, 51, 55–57, 60, 76,	<code>\pcol@bg@@n</code> 90, 169, 172		
.....	76, 78, 90–92, 94, 95, 101, 102, 104–	<code>\pcol@bg@@P</code> 172		
.....	108, 113, 114, 117–119, 125, 127,	<code>\pcol@bg@@p</code> 90, 169, 172		
.....	133–139, 145, 154, 156–161, 163,	<code>\pcol@bg@@paintbox</code>	.. 161, 166 , 166, 167, 230		
.....	165–168, 170, 172, 176, 214, 216–	<code>\pcol@bg@@paintcolumns</code>	166 , 166, 167, 230		
.....	219, 221, 224, 225, 230, 258, 267, 268	<code>\pcol@bg@@paintpage</code>		
<code>\backgroundcolor</code> 27,	129, 161, 166 , 166, 167, 230		
.....	27, 28, 48 , 48, 52 , 52, 55, 56 , 60, 76,	<code>\pcol@bg@@R</code> 172		
.....	77, 105, 108, 123, 136, 265 , 265, 267, 268	<code>\pcol@bg@@r</code> 90, 169, 172		
<code>\pcol@backgroundcolor</code>	<code>\pcol@bg@@S</code> 172		
.....	108, 110, 123, 265 , 265, 267, 268	<code>\pcol@bg@@s</code> 90, 172		
		<code>\ifpcol@bg@@swap</code> 85, 135 , 136, 167		

`\pcol@bg@T` [172](#)
`\pcol@bg@t` [169](#), [172](#)
`\pcol@bg@addext` [94](#), [109](#), [125](#), [168](#), [169](#), [169](#)
`\pcol@bg@advance` [119](#), [169](#), [169](#)
`\pcol@bg@calculate` [119](#), [168](#), [169](#), [169](#)
`\pcol@bg@color@a` [76](#),
[110](#), [114](#), [120](#), [125](#), [168](#), [265](#), [267](#), [268](#)
`\pcol@bg@color@a@c` [76](#),
[110](#), [114](#), [120](#), [125](#), [168](#), [265](#), [267](#), [268](#)
`\pcol@bg@color@xx` [267](#), [267](#)
`\pcol@bg@columnheight`
[118](#), [119](#), [159](#), [171](#), [171](#), [172](#)
`\pcol@bg@columnleft`
[86](#), [109](#), [111](#), [128](#), [165](#), [170](#), [170](#)
`\pcol@bg@columnright` [170](#), [170](#)
`\pcol@bg@columnsep` [110](#), [128](#), [170](#), [170](#)
`\pcol@bg@columntop` [159](#), [171](#), [171](#), [172](#)
`\pcol@bg@columnwidth` .. [110](#), [128](#), [170](#), [170](#)
`\pcol@bg@defext` [96](#), [265](#), [266](#), [267](#), [267](#)
`\pcol@bg@defined` .. [113](#), [120](#), [265](#), [268](#), [268](#)
`\pcol@bg@dimen` [169](#), [169](#)
`\pcol@bg@ext@d@a` [77](#), [94](#), [109](#), [125](#), [169](#), [268](#)
`\pcol@bg@ext@d@a@c`
[77](#), [94](#), [109](#), [125](#), [169](#), [268](#)
`\pcol@bg@ext@inf@d` [169](#)
`\pcol@bg@ext@inf@b` [169](#), [170](#)
`\pcol@bg@ext@inf@l` [169](#), [169](#), [170](#)
`\pcol@bg@ext@inf@r` [169](#), [169](#), [170](#)
`\pcol@bg@ext@inf@t` [169](#), [169](#), [170](#)
`\pcol@bg@floatheight`
[118](#), [119](#), [154](#), [156](#), [171](#), [171](#), [172](#), [216](#)
`\pcol@bg@footnoteheight` [118](#),
[119](#), [125](#), [156](#), [171](#), [171](#), [172](#), [219](#), [225](#)
`\pcol@bg@from` [129](#), [158](#), [161](#), [165](#), [165](#)
`\pcol@bg@leftmargin`
[85](#), [137](#), [137](#), [166](#), [167](#), [170](#)
`\pcol@bg@mayhavecol@` [265](#)
`\pcol@bg@mayhavecol@a` [267](#)
`\pcol@bg@mayhavecol@C` [265](#), [267](#)
`\pcol@bg@mayhavecol@c` [265](#), [267](#)
`\pcol@bg@mayhavecol@G` [265](#), [267](#)
`\pcol@bg@mayhavecol@g` [265](#), [267](#)
`\pcol@bg@nadvance` [119](#), [169](#), [169](#)
`\pcol@bg@negative` [119](#), [169](#), [169](#)
`\pcol@bg@pageleft` . [137](#), [138](#), [169](#), [170](#), [170](#)
`\pcol@bg@pagetop` [89](#), [90](#), [138](#), [169](#), [170](#), [170](#)
`\pcol@bg@paint@i` [83](#), [91](#), [101](#), [105](#), [106](#),
[129](#), [136–138](#), [165](#), [166](#), [166–168](#), [170](#)
`\pcol@bg@paint@ii` [82](#), [111](#), [112](#),
[127](#), [129](#), [135](#), [165](#), [166](#), [167](#), [167](#), [168](#)
`\pcol@bg@paintbox`
[110](#), [125](#), [129](#), [154](#), [156](#), [158](#), [159](#),
[166](#), [166](#), [171](#), [176](#), [216](#), [219](#), [225](#), [230](#)

`\pcol@bg@paintcolumns`
[129](#), [159](#), [166](#), [166](#), [171](#), [230](#)
`\ifpcol@bg@painted` [104](#), [136](#), [166](#), [168](#)
`\pcol@bg@paintpage`
[129](#), [161](#), [166](#), [166](#), [171](#), [230](#)
`\pcol@bg@paintregion`
[91](#), [101](#), [110](#), [125](#), [136](#), [167](#), [168](#), [168](#)
`\pcol@bg@paintregion@i` ... [94–97](#), [108](#),
[123](#), [125](#), [136](#), [168](#), [168](#), [169](#), [172](#), [262](#)
`\pcol@bg@paperheight` ... [91](#), [138](#), [170](#), [170](#)
`\pcol@bg@paperwidth` [91](#), [138](#), [170](#), [170](#)
`\pcol@bg@preposttop`
[95](#), [161](#), [171](#), [171](#), [172](#), [176](#)
`\pcol@bg@preposttop@left`
[161](#), [171](#), [171](#), [176](#), [224](#)
`\pcol@bg@preposttop@right`
[161](#), [171](#), [171](#), [176](#), [224](#)
`\pcol@bg@region` [265](#), [265](#), [267](#)
`\pcol@bg@spanningheight`
[119](#), [158](#), [171](#), [171](#), [172](#)
`\pcol@bg@spanningtop` [119](#), [158](#), [171](#), [171](#), [172](#)
`\ifpcol@bg@swap` [135](#), [136](#), [167](#), [258](#)
`\pcol@bg@swappage`
[85](#), [89–91](#), [102](#), [136](#), [137](#), [167](#), [167](#)
`\pcol@bg@textheight` [90](#),
[92](#), [119](#), [125](#), [154](#), [170](#), [170](#), [176](#), [225](#)
`\pcol@bg@to` [129](#), [158](#), [161](#), [165](#), [165](#)
`\pcol@bias@mpbout`
[125](#), [176](#), [199](#), [199](#), [200](#), [223](#)
`\pcol@bias@mpbout@i`
[94](#), [119](#), [122](#), [125](#), [127](#), [199](#), [200](#)
`\bigskip` [55](#)
`\botfigrule` [107](#), [210](#)
`\@botlist` [65](#), [117](#), [119](#), [120](#), [124](#), [165](#), [205](#), [206](#)
`\@botnum` [65](#), [84](#)
`\@botroom` [65](#), [93](#)
`\bottomfraction` [178](#)
`\box` [65](#), [66](#), [75](#), [81](#), [88](#), [92](#),
[95](#), [96](#), [99–101](#), [116](#), [138–140](#), [145](#),
[146](#), [157](#), [179](#), [186](#), [192](#), [204](#), [205](#), [219](#)
`\boxmaxdepth` [88](#), [92](#), [145](#), [148](#), [158](#), [214](#)
`\@bsphack` [114](#)
`\pcol@buildcolseprule` [66](#), [90–92](#),
[94–96](#), [101](#), [108](#), [118](#), [129](#), [139](#), [157](#),
[158](#), [158](#), [160](#), [165](#), [166](#), [171](#), [219–221](#)
`\pcol@buildcselt` .. [90](#), [91](#), [94–97](#), [101](#),
[108](#), [118](#), [119](#), [139](#), [158](#), [159](#), [166](#), [171](#)
`\pcol@buildcselt@S`
[91](#), [94](#), [95](#), [101](#), [118](#), [119](#), [158](#), [158](#)
`\bx@A` [81](#), [81](#)
`\bx@AA` [81](#)
`\bx@R` [81](#)
`\bx@Z` [81](#)
`\bx@ZZ` [81](#)

C

<p><i>C</i> 64, 64, 67–69, 74–76, 78, 79, 84, 86, 87, 90, 102–105, 111, 127– 130, 135, 150, 154–156, 161, 165, 176, 177, 182, 195, 201–204, 209, 211, 214, 215, 217, 219–222, 224, 227, 229, 235, 242, 244–246, 249, 259</p> <p>C^0 83, 85–87, 95, 97, 111, 126, 128, 129, 130, 156–160, 195, 217–219, 221, 229, 235–237, 239, 259</p> <p>C^1 83, 85–87, 95, 97, 111, 126, 128, 129, 130, 156–160, 167, 195, 217–219, 221, 229, 235–237, 239, 259</p> <p>C_b^0 83, 86, 111, 129, 158, 161, 165, 165–168, 170</p> <p>C_b^1 83, 129, 158, 161, 165, 165–168</p> <p>C_L 74, 74, 75, 77, 86, 87, 90, 111, 121, 125, 127, 129, 129, 135, 154–156, 160, 161, 165, 176, 194, 195, 197, 198, 214, 215, 217, 219, 220, 224, 227, 229, 235, 259</p> <p>c_{\max} 202, 202, 203</p> <p>$\backslash c@{\theta}$ 85, 110, 244</p> <p>$\backslash c@botnumber$ 177</p> <p>$\backslash c@footnote$ 83, 85, 130, 134, 231, 256, 257, 268</p> <p>$\backslash c@page$ 66, 84, 85, 135, 142, 149–154, 156, 162, 176, 181, 183, 221, 225, 242, 259</p> <p>$\backslash c@topnumber$ 177</p> <p>$\backslash c@totalnumber$ 177</p> <p>$\backslash pcol@calcfncntr$ 85, 87, 123, 126, 127, 130, 256, 256, 257</p> <p>$\backslash caption$ 11, 115</p> <p>$\backslash @caption$ 115, 252</p> <p>$\backslash catcode$ 240, 241</p> <p>$\backslash @cclv$ 99, 175</p> <p>$\backslash pcol@ccuse$ 110, 128, 191, 191</p> <p>$\backslash pcol@ccxdef$ 128, 191, 191</p> <p>$\backslash @cdr$ 113, 122</p> <p>$\backslash @cflb$ 117, 124, 147</p> <p>$\backslash @cflt$ 88, 117, 147, 148, 213</p> <p>$\backslash pcol@cflt$ 84, 88, 92, 98, 100, 101, 107, 117, 118, 121, 122, 124, 137, 147, 148, 148, 212, 213</p> <p>$\backslash chardef$ 82, 83, 127</p> <p>$\backslash pcol@checkshipped$ 82, 86, 91, 104, 111, 120, 129, 149, 150, 150</p> <p>$\backslash cl@{\theta}$ 85, 109, 110, 124, 126, 228, 242, 242, 245</p> <p>$\backslash pcol@cl@{\theta}$ 68, 115, 126, 242</p> <p>$\backslash cl@@ckpt$ 68, 124, 228, 242</p> <p>$\backslash ifpcol@clear$ 104, 132, 182, 201, 203, 206, 211, 215, 253</p> <p>$\backslash pcol@clearcolorstack$ 71, 92, 114, 125, 127, 128, 140, 188, 188, 189</p>	<p>$\backslash pcol@clearcst@unvbox$ 71, 139, 148, 180, 187, 188, 188, 189</p> <p>$\backslash cleardoublepage$ 21, 29, 29, 64, 102, 233, 252, 252</p> <p>$\backslash clearpage$ 12, 16, 29, 29, 64, 116, 137, 207, 214, 222, 227, 233, 252, 252</p> <p>$\backslash clubpenalty$ 19</p> <p>$\backslash pcol@cmpctrelt$ 85, 86, 105, 110, 113, 120, 128, 243, 244, 244</p> <p>$\backslash pcol@col@c$ 65, 185</p> <p>$\backslash col@number$ 84, 229, 247</p> <p>$\backslash @colht$ 66, 69, 81, 90, 92, 92, 93, 95, 133, 137, 145–147, 150–153, 164, 176–178, 183, 192, 193, 204–206, 208, 211, 214, 215, 217, 219–222, 226</p> <p>$\backslash pcol@colht$ 133, 137, 137, 202, 215, 218, 222</p> <p>$\backslash @colnum$ 65, 84</p> <p>$\backslash color$ 26, 70, 72–74, 107, 114, 260–262, 264, 267</p> <p>color context 70, 70, 139, 143, 148, 164, 178, 180, 183, 187–189, 191</p> <p>color stack 70, 71–74, 130, 131, 134, 139, 140, 143, 148, 180, 186–188, 225, 230, 261, 262</p> <p>$\backslash color@begingroup$ 70, 114</p> <p>$\backslash color@endgroup$ 70, 114</p> <p>$\backslash pcol@colorcommand$ 260, 260, 264, 264</p> <p>$\backslash coloredwordhyphenated$ 26, 26, 262, 264, 264, 270</p> <p>$\backslash pcol@colorins$ 71, 86, 88, 91, 140, 140, 188, 189, 191, 262, 263</p> <p>$\backslash pcol@colorstack@savd$ 74, 86, 139, 139, 140, 187–189, 191</p> <p>$\backslash pcol@colorstack@shadow$ 72, 120, 230, 261, 262</p> <p>$\backslash pcol@colpream@c$ 110, 132, 232, 248, 250</p> <p>$\backslash @colroom$ 65, 87, 93, 93, 97, 132, 137, 144, 164, 174, 175, 177, 178, 180, 181, 183, 187, 206, 207, 222, 226</p> <p>$\backslash pcol@colsepid$ 157, 195, 220, 259, 259</p> <p>$\backslash colseprulecolor$ 26, 26, 27, 76, 108, 110, 160, 264, 264</p> <p>$\backslash pcol@colseprulecolor$ 107, 109, 160, 264, 264</p> <p>$\backslash pcol@colseprulecolor@c$ 109, 264, 264</p> <p>$\backslash column$ 232, 233, 249, 249</p> <p>column (environment) 7, 8, 16, 17, 17, 232, 249, 249, 250</p> <p>column preamble 18, 18, 232, 250</p> <p>$\backslash column*$ 109, 110, 249</p> <p>column* (environment) 8, 17, 17, 19, 249</p> <p>column-context 64, 65, 75, 76, 87, 89, 181–183, 186, 204, 206, 209, 211, 214, 219–222, 249</p>
--	--

column-page 64, 64–74, 81, 82, 84, 87–90, 92, 93, 95, 97–101, 104, 107, 109, 112, 113, 116, 117, 122, 124, 130, 133, 137, 139, 140, 143–149, 151, 158, 164, 175–181, 183, 184, 186–188, 192, 193, 201–203, 205, 207–209, 211–215, 217–220, 222, 224, 248, 249
column-scan 69, 87, 107, 108, 131, 132, 180, 182, 187, 217, 227, 248, 249, 252, 253
column-separating rule 26, 66, 76, 76–78, 83, 86, 90, 92, 101, 107, 108, 135, 136, 139, 145, 156–159, 165, 219–221, 229, 239, 259, 264
column-swapping 21, 22, 42, 75, 75, 76, 78, 85, 87, 90, 95, 129, 132, 135, 145, 157, 160–162, 180, 182, 219–221, 228, 258, 259
column-switching 5, 11, 16, 17, 17–19, 23, 26, 59, 60, 67–71, 76, 86, 87, 107, 108, 131, 143, 180, 182, 187, 201, 203, 212, 227, 232, 246–248, 250, 251, 260, 262, 263
column-switching environment 7, 16–18, 60, 64, 108, 227
column-wise footnote 24, 24, 31, 34, 59, 65, 68, 70, 81, 87, 90, 99, 113, 133, 145, 146, 177, 181, 184, 192, 193, 219, 220, 231, 253, 268
column-wise stuff 15, 16, 29, 61, 65, 79, 107, 115, 117, 118, 123, 124, 133, 136, 151, 175, 177, 210, 214, 220, 223, 226
\columncolor 25, 25, 26, 71, 72, 108, 110, 128, 190, 260, 260, 262
\pcol@columncolor 110, 128, 260, 260
\pcol@columncolor·c 72, 110, 128, 178, 191, 260, 261
\pcol@columncolor@box·c 71, 72, 110, 128, 178, 191, 260
\columnratio 12, 19, 20, 21, 38, 41, 44, 77, 86, 87, 97, 110–112, 126, 234, 234, 235
\pcol@columnratioleft 111, 126, 229, 234, 234, 235
\pcol@columnratoright 111, 229, 234, 234, 235
\columnsep . . 15, 19–21, 90, 95, 97, 234–237
\pcol@columnsep·c 77, 95, 109, 110, 157, 160, 170, 195, 229, 235, 236, 238
\columnseprule 27, 90, 91, 159, 160
\columnwidth 20, 66, 89, 90, 90, 107, 109, 134, 145, 182, 185, 220, 221, 231, 247, 255, 268
\pcol@columnwidth·c . . 66, 77, 109, 110, 157, 170, 185, 195, 229, 235, 236, 238
\pcol@colwidthspecleft . 229, 234, 234–236
\pcol@colwidthspecright 229, 234, 234–236
\pcol@com@⟨com⟩ 232, 233
\pcol@com@cleardoublepage . . 102, 252, 252
\pcol@com@clearpage 100, 173, 227, 252, 252, 253
\pcol@com@column 232, 246, 249, 249
\pcol@com@column* 109, 110, 232, 249
\pcol@com@endcolumn . . . 102, 246, 250, 250
\pcol@com@endcolumn* 250, 250
\pcol@com@endleftcolumn 250, 250
\pcol@com@endleftcolumn* 250
\pcol@com@endnthcolumn 250, 250
\pcol@com@endnthcolumn* 250
\pcol@com@endrightcolumn 250, 250
\pcol@com@endrightcolumn* 250
\pcol@com@flushpage 100, 173, 227, 252, 252, 253
\pcol@com@leftcolumn 249, 249
\pcol@com@leftcolumn* 109, 249
\pcol@com@nthcolumn 249, 249
\pcol@com@nthcolumn* 109, 249
\pcol@com@rightcolumn 249
\pcol@com@rightcolumn* 109, 249
\pcol@com@switchcolumn 83, 86, 92, 110, 128, 130, 246, 246, 249
\pcol@com@syncallcounters 83, 92, 111, 128, 129, 244, 245, 245
\pcol@com@synccounter 120, 244, 244
\@combinefloats 117, 143, 146, 146, 147, 177, 179, 226, 229
\pcol@combinefloats 88, 91, 92, 98, 100, 108, 109, 117, 120, 124, 133, 137, 143, 146, 146–148, 175, 179, 229
\pcol@@combinefloats 117, 143, 146, 147, 229
\pcol@combinefootins 91, 92, 100, 138, 179, 193, 193
\@comdblflleft 101, 117, 118, 153
\@comflelt 101, 117, 118
\@cons 112, 148, 154–157, 178, 183, 205, 206, 208, 233, 242–244
\copy 184, 186
\count 18, 65, 71, 82–85, 87, 91, 128, 140, 186, 192, 259
\count@ 85, 241
counter synchronization 85, 243, 244
\pcol@counters 68, 68, 228, 242, 244
\pcol@counters·c 68, 110, 244
counters:
figure 10, 79
footnote . 10, 24, 25, 30–33, 85, 106, 107, 116, 130, 242, 253, 255, 256, 268
page 11, 16, 23, 24, 42, 66, 67, 84, 85, 106, 124, 149, 228, 242

section 11
 subsection 10, 11
 table 10, 79
 \csname 185, 242, 243, 245
 \pcol@ctr@ θ 109, 110, 229, 243, 244
 \@currbox . 113, 121, 121, 187, 191, 195, 222
 \pcol@currboxsave 222, 222
 \pcol@currcol 87, 91, 128, 128, 129, 132,
 142, 178, 180, 183, 185, 186, 233, 253
 current column-page 64, 65, 81, 84, 86, 112,
 113, 124, 130, 139, 143, 145, 180,
 182, 183, 191, 201, 203–207, 209, 211
 \current@color
 . . 72, 114, 123, 168, 188, 261, 262, 267
 \@currentlabel 113, 116, 121
 \pcol@currfoot 65,
 99, 133, 146, 152, 181, 183, 186, 192
 \@currlist 112, 122, 128, 195
 \pcol@currpage
 . . . 68, 68, 113, 118, 149–152, 175, 222

D

d_c 202, 202, 210, 211
 D_P 89, 93,
 94, 104, 136, 202, 202, 210, 211, 223, 225
 D_T 89, 94, 97, 104, 202, 202–204, 209, 211–213
 \@dbldeferlist 79, 80, 103, 113, 116–119,
 121, 123, 123, 124, 126, 152, 153,
 175, 214, 215, 223, 224, 226, 233, 233
 \dblfigrule 107, 153
 \@dblfloatplacement . 79, 116, 124, 152, 214
 \dblfloatsep 98, 153
 \dbltextfloatsep . . . 55, 98, 153, 216, 223
 \@dbltoplist
 . 101, 117, 118, 120–122, 123, 153, 215
 \deadcycles 81, 226
 \def .. 68, 72, 73, 79, 105, 108, 109, 112–
 116, 118–120, 123–125, 127, 140,
 145, 176, 186–189, 192, 197, 200,
 219, 222, 223, 225, 232–234, 240–
 243, 246, 250, 259–261, 264, 265, 267
 \pcol@def@extract@fil
 105, 123, 240, 241, 241
 \pcol@def@extract@fil@iii
 94, 123, 240, 241, 241
 \pcol@def@extract@pt 240, 241, 241
 \pcol@defcolumn . . . 109, 232, 246, 249, 249
 \pcol@defcomelt . . . 120, 126, 232, 233, 233
 \pcol@defcseprulecolor . . . 110, 264, 264
 \pcol@defcseprulecolor@i
 105, 108, 109, 114, 264, 264
 \pcol@defcseprulecolor@x . . . 107, 264, 264
 \pcol@defcseprulecolor@y . . . 107, 264, 264

\pcol@defcurrpage 68, 86, 118,
 145, 149, 150, 150, 153, 176, 180, 198
 \@deferlist 65, 79, 80, 117–119, 121, 123,
 123, 127, 142, 153, 165, 175, 206, 226
 \pcol@deferredfootins
 . 69, 81–83, 88, 91–93, 96, 98, 101,
 133, 138, 164, 183, 192, 192, 225, 254
 \definecolumnpreamble 18, 18, 87, 250, 250
 \definethecounter
 . . . 12, 24, 68, 106, 109, 228, 243, 243
 \pcol@defkw 92, 99, 109, 123, 240, 240
 description (environment) 15
 \ifpcol@dfloats 104, 105,
 124, 133, 136, 202, 210, 215–217, 223
 \dimen 65, 66, 88–97, 137,
 138, 140, 166, 169, 185, 186, 192, 259
 \dimen@ 94, 168, 210, 236, 238, 239
 \dimen@ii 94, 236, 238, 239, 241
 \pcol@do@mpb@all 199, 200, 200
 \pcol@do@mpb@all@i 122, 199, 200, 200
 \pcol@do@mpb@all@ii
 122, 125, 127, 199, 200, 200
 \pcol@do@mpbout 176, 199, 199, 223
 \pcol@do@mpbout@elem
 85, 119, 176, 199, 199, 223
 \pcol@do@mpbout@i
 83, 85, 86, 91, 103, 199, 199
 \pcol@do@mpbout@whole . 176, 199, 199, 223
 \@doclearpage 160, 215
 document (environment) 87, 92
 \documentclass 21, 22, 28, 29, 38, 77
 \dp 78
 \pcol@dummysymbol 140, 230, 230, 269

E

$e_a(d^\pm)$ 77, 77
 \edef 73, 113, 118–120, 183, 216, 262
 \@eha 113
 \@ehb 113, 142
 \@elt 64, 66–68, 72, 112, 113, 118,
 118–120, 122, 127, 153, 169, 176,
 199, 200, 232, 244, 248, 260–262, 268
 \pcol@elt@save 262
 \@empty 120, 120, 121, 128
 \@emptycol 116
 \end 7, 8, 11, 15–18, 25,
 33, 55, 57, 64, 72, 75, 94, 107, 116,
 117, 124, 205, 225, 230, 256, 268, 269
 \end@dblfloat . . . 79, 80, 102, 115, 229, 233
 \pcol@end@dblfloat 79,
 80, 84, 102, 113–115, 122, 229, 233, 233
 \end@float 115, 226
 \endcolumn 233, 250, 250
 \endcolumn* 250, 250

<code>\endcsname</code>	185, 242, 243, 245
<code>\@endfloatbox</code>	79, 115
<code>\endgroup</code>	73, 229
<code>\endleftcolumn</code>	233, 250, 250
<code>\endleftcolumn*</code>	250
<code>\endnthcolumn</code>	233, 250, 250
<code>\endnthcolumn*</code>	250
<code>\endparacol</code>	85, 90, 92, 97, 100, 102, 129, 130, 132–134, 138, 174, 222, 227, 246, 248, 253, 268, 268
<code>\endrightcolumn</code>	233, 250, 250
<code>\endrightcolumn*</code>	250
<code>\enlargepage</code>	205
<code>\ensurevspace</code>	19, 19, 96, 97, 132, 203, 248, 250, 250, 270
<code>\pcol@ensurevspace</code>	96, 97, 248, 250, 250
<code>\pcol@ensurevspace</code>	96, 203, 248, 250, 250
<code>enumerate (environment)</code>	15
<code>environment-local</code>	232, 233, 233, 244–246, 249, 250, 252
<code>environments:</code>	
<code>column</code>	7, 8, 16, 17, 17, 232, 249, 249, 250
<code>column*</code>	8, 17, 17, 19, 249
<code>description</code>	15
<code>document</code>	87, 92
<code>enumerate</code>	15
<code>figure</code>	9, 10, 57, 59, 60
<code>figure*</code>	9, 57, 60
<code>itemize</code>	15
<code>leftcolumn</code>	8, 9, 16, 18, 18, 249, 249
<code>leftcolumn*</code>	8, 10, 18, 18, 19, 109, 249
<code>list</code>	15, 16, 20, 84, 89, 92, 98, 103, 106, 115, 227–229, 247
<code>minipage</code>	116
<code>nthcolumn</code>	8, 8, 16, 17, 17, 18, 249, 249
<code>nthcolumn*</code>	8, 17, 17, 18, 109, 249
<code>paracol</code>	6, 6, 7, 10–12, 15, 15, 16, 16, 18–27, 29–35, 38, 39, 41, 42, 44, 48, 50, 51, 54–57, 60, 61, 64, 66, 68, 70, 72, 74, 75, 78–81, 83–87, 89, 90, 92, 94, 97, 98, 101–103, 106–108, 115– 117, 123–125, 128–132, 134, 135, 138–140, 143, 147, 148, 152, 155, 160, 161, 163, 166, 171, 174–178, 188, 193, 194, 198, 199, 205, 223– 226, 227, 227–234, 246, 247, 249, 253, 256, 260, 262, 263, 268, 269
<code>rightcolumn</code>	8, 10, 16, 18, 18, 249, 249
<code>rightcolumn*</code>	8, 18, 18, 109, 249
<code>table</code>	9, 11, 57, 59
<code>table*</code>	9, 57
<code>trivlist</code>	103, 228
<code>verse</code>	19
<code>\@Esphack</code>	114
<code>\evensidemargin</code>	21, 38, 77, 89, 89, 92, 102, 162, 167
<code>\everyhbox</code>	72, 73
<code>\everypar</code>	60, 66, 102, 102, 115, 132, 178, 181, 182, 225, 247, 250
<code>\everyvbox</code>	72, 73, 102, 134, 139, 140, 163, 230, 269
<code>\pcol@everyvbox</code>	102, 140, 140, 230, 269
<code>\expandafter</code>	168, 185, 197, 199, 200, 233, 237, 242, 243, 245, 247, 257
<code>explicit synchronization</code>	64, 64, 69, 74, 201, 203, 246
<code>\ext@figure</code>	110, 121, 252
<code>\ext@table</code>	110, 121, 252
<code>extension of background painting region</code>	27, 28, 28, 77, 94, 157, 168, 169, 265, 266
<code>\pcol@extract@fil</code>	105, 123, 238, 240, 241, 241
<code>\pcol@extract@fil@i</code>	85, 123, 126, 241, 241
<code>\pcol@extract@fil@ii</code>	91, 97, 123, 126, 127, 240, 241, 241
<code>\pcol@extract@fil@iii</code>	97, 123, 240, 241, 241
<code>\pcol@extract@pt</code>	239, 240, 241, 241
F	
<code>\pcol@F</code>	110, 141, 142
<code>$F_c(X)$</code>	201, 202, 203, 209–211
<code>$f_c(x)$</code>	201, 201, 202, 209, 210
<code>\pcol@F@count</code>	82, 85, 91, 118, 122, 141, 142
<code>\f@depth</code>	79, 80, 116, 124, 152, 214
<code>\pcol@F@n</code>	125, 141, 142
<code>\pcol@F@write</code>	141, 142
<code>false</code>	65, 72, 73, 82, 83, 102–106, 131–136, 143, 144, 150, 151, 154–156, 161, 166, 167, 175, 182, 187, 190, 194, 195, 199, 202–204, 210, 211, 215– 217, 219–221, 223–230, 247, 248, 252, 253, 255, 258, 259, 263, 265, 268
<code>\pcol@Fb</code>	141, 142
<code>\if@fcolmade</code>	103, 105, 111, 152, 165, 215, 220, 221, 223
<code>\pcol@fcwhyphenate</code>	262, 264, 264
<code>\pcol@Fe</code>	110, 125, 141, 142
<code>\pcol@FF</code>	85, 125, 128, 130, 141, 142
<code>figure (counter)</code>	10, 79
<code>figure (environment)</code>	9, 10, 57, 59, 60
<code>figure* (environment)</code>	9, 57, 60
<code>\fill</code>	20, 21, 98, 98, 121, 127, 128, 236–238
<code>\@finalstrut</code>	115
<code>\if@firstcolumn</code>	102, 136, 194–198, 215, 229
<code>\ifpcol@firstpage</code>	134, 155, 175, 176, 216, 217
<code>\pcol@firstprevdepth</code>	88, 152, 223, 224, 229

float column 108, 116, 133, 134, 138, 146, 164,
16, 29, 89, 92–95, 98–101, 103–105,
107, 111–113, 117–119, 123, 124,
137, 148, 201, 205–208, 215, 219–222

float page 29, 66, 85, 88, 89, 95,
99, 100, 103, 104, 111, 112, 116, 117,
130, 132, 133, 149, 150, 152–155,
175, 201, 204, 207, 214, 215, 223, 226

\pcol@float 212

\floatingpenalty 82, 84, 255

\floatpagefraction 93, 207

\@floatpenalty 84, 84, 257

\@floatplacement 115, 151, 177, 178

\pcol@floatplacement 88, 91, 94, 115,
137, 149, 151, 151, 177, 206, 222, 226

\floatsep 96,
97, 98, 137, 148, 201, 207, 210, 212

\@fltovf 142

\ifpcol@flush
. 100, 111, 132, 187, 203, 248, 253

\flushbottom 17, 59, 61, 145

\pcol@flushclear 84, 100, 108, 111,
128, 129, 131–134, 138, 173, 180,
203, 227, 246, 249, 252, 253, 253, 268

\pcol@flushcolumn 69, 81, 83, 86, 89,
92, 93, 99, 100, 103, 111–113, 116,
118, 120–124, 128, 130, 132, 133,
142, 144, 147, 150, 151, 165, 185,
186, 192, 201, 203, 204, 204, 207, 208

\pcol@flushfloats 90–
92, 100, 101, 103, 111, 128, 129,
139, 160, 214, 215, 220, 220, 221, 223

\flushpage 12, 12, 16, 29, 29, 48,
64, 137, 207, 214, 222, 227, 233, 252, 252

\fncounteradjustment
. 24, 25, 25, 33, 134, 253, 255, 255

\ifpcol@fncounteradjustment
. 134, 134, 253, 255, 268

\pcol@fnheight@lpage 218, 225

\pcol@fnlayout@l 110, 253

\pcol@fnlayout@c
. 110, 133, 134, 242, 253, 253, 254

\pcol@fnlayout@m 110, 133, 134, 253, 253, 254

\pcol@fnlayout@p
. 110, 133, 134, 242, 253, 253, 254

\pcol@fntext 116, 126, 130, 134, 231, 254, 254

\pcol@fntextbody
81, 82, 84, 87–91, 96–99, 101, 106,
108, 113–116, 121, 134, 254, 255, 255

\pcol@fntextother
. 82, 126, 134, 138, 254, 254, 255

\pcol@fntexttop 81, 82, 126, 254, 254, 255

\footins 55,
68, 69, 78, 80, 81, 86, 97, 99, 101,
108, 116, 133, 134, 138, 146, 164,
172, 175, 178–181, 183, 184, 186,
192–194, 204, 205, 219, 225, 254, 255

\pcol@footins 66, 150, 151, 153, 154, 164, 192

\footnote 10, 25, 25, 31, 32, 32–37, 60, 68,
69, 80, 107, 116, 130, 231, 254, 256, 256

footnote (counter)
. 10, 24, 25, 30–33, 85, 106,
107, 116, 130, 242, 253, 255, 256, 268

\pcol@footnote 107, 111, 231, 256, 256

\pcol@@footnote 107, 256, 256

\pcol@footnotebase 85, 130, 130, 231, 256

\footnotelayout
. 24, 108, 110, 231, 233, 253, 254

\footnotemark 25, 25, 31, 32,
32, 33, 35, 36, 107, 130, 231, 256, 256

\pcol@footnotemark 107, 111, 231, 256, 256

\pcol@@footnotemark 107, 256, 256

\footnoteplacement 24, 24, 25, 30,
33, 57, 108, 110, 231, 233, 253, 253, 254

\footnoterule 90, 107, 134, 209, 231, 256

\pcol@footnoterule 107, 231, 256, 256

\footnotesep 89, 98, 106

\footnotesize 106

\footnotetext 25,
25, 31, 32, 32, 34–36, 68, 69, 80,
107, 116, 130, 231, 254, 256, 257, 257

\@footnotetext 116, 231, 254, 254, 255

\pcol@footnotetext 107, 111, 231, 257, 257

\pcol@@footnotetext 107, 256, 256, 257

\footskip 55

\@for 112, 126, 235, 237

\@fpbot 99, 208

\@fpmin 93, 104, 207

\@fpsep 97, 98, 207, 208

\@fptop 98, 208

\@freelist 72, 73, 85, 100,
108, 112, 113, 118, 119, 122, 122–
125, 142, 148, 152–154, 156–158,
164, 176, 178–180, 183, 184, 189,
191, 192, 204, 208, 212, 219, 223, 225

\pcol@freshpage
. 68, 83, 91, 93, 103, 111, 121,
122, 128–130, 137, 143, 148, 150–
152, 164, 183, 185, 186, 214, 221, 221

G

g_c 20, 77, 95,
96, 108, 157, 160, 195, 229, 234–237, 239

\pcol@gcounters 68, 120, 228, 242, 242

\gdef 238, 242

\pcol@getcurrcol 90,
128, 177, 178, 182, 183, 185, 185,
186, 204, 209, 211, 219, 221, 222, 225

`\pcol@getcurrfoot` 81,
 99, 164, 183, 184, 186, 186, 204, 219
`\pcol@getcurrpage`
 85, 86, 118, 130, 149, 150,
 150, 156, 180, 183, 194, 204, 206, 222
`\pcol@getcurrpinf`
 85, 86, 95, 113, 122, 125, 145,
 150, 151, 152, 156, 180, 201, 206, 215
`\pcol@getmparbottom` 67, 91, 94–97, 104,
 113, 119–121, 125, 195, 197, 197, 198
`\pcol@getmparbottom@i`
 102, 125, 128, 129, 197, 197
`\pcol@getmparbottom@last`
 120, 122, 125, 198, 199, 200, 200, 222
`\pcol@getmparbottom@last@i`
 119, 122, 125, 127, 200, 200
`\pcol@getmpbelt`
 94–97, 104, 113, 119, 197, 197
`\pcol@getpelt`
 83, 85, 86, 91, 92, 97, 118, 150, 151
`\pcol@getpinf`
 90, 125, 131, 138, 150, 151, 154
`\global` 102, 115, 127–132, 134–
 136, 139, 140, 151, 155, 156, 161,
 166, 182, 185, 187, 217, 227, 228,
 230, 234, 242, 245, 247, 250, 262, 269
`global counter` 10,
 11, 23, 23, 68, 68, 113, 115, 242, 245
`\globalcounter` 10, 10, 11, 23, 23, 68, 105,
 111, 113, 120, 126, 127, 242, 242, 270
`\pcol@globalcounter` 242, 242
`\pcol@globalcounter@es` 124, 242, 242
`\@gobble` 110, 110, 142, 166, 230
`\pcol@gobblethree` 106, 251, 251, 252
`\@gtempa` 105,
 121, 127, 187, 229, 237, 238, 243, 244

H

`\H@@footnotetext` 231
`HB` 78, 78, 161, 176
`HM` 77, 77, 78, 169, 170, 172
`HN` 78, 78
`HP` 77, 77, 78, 91, 169, 170, 172
`HR` 77, 77, 78, 169, 170, 172
`HS` 77, 77, 78
`HT` ... 78, 78, 90, 92, 154, 161, 166, 170, 172
`\ifpcol@havelastpage` 134, 161, 224
`\hb@xt@` 109
`\hbox` 72, 73, 90,
 95, 109, 157, 168, 187, 199, 219–221, 252
`\pcol@hdfilelt` 94, 98, 119, 210, 210
`\headheight` 77, 89, 89, 90, 95, 160, 170
`\headsep` 55, 77, 89, 90, 95, 160, 170
`\@height` 108

`\hfil` 239
`\pcol@hfil` 90,
 95, 108, 139, 157, 160, 160, 220, 221, 264
`\pcol@hfil` 157, 220, 221
`\@holdpg` 65,
 93, 96, 100, 174, 175, 179, 180, 226
`\hrule` 108, 177, 187
`\hsize` 20, 87, 90, 134, 227, 247, 255
`\hskip` 262, 264
`\hss` 157
`\ht` 78, 193, 222

I

`\pcol@iadjustfnctr` 83, 85, 87, 123, 256, 256
`\pcol@icolumncolor`
 91, 101, 103, 105, 106, 108,
 110, 112, 114, 120, 122, 123, 128,
 134, 140, 142, 191, 232, 260, 260, 262
`\pcol@icolumnratio` 234, 234
`\pcol@ifccdefined` 128, 191, 191
`\pcol@ifempty` 74,
 101, 125, 127, 139, 187, 187, 188, 209
`\ifhmode` 72, 73
`\ifinner` 72, 73, 114, 227
`\pcol@iflushfloats` 83,
 85, 87, 90–92, 101, 103, 105, 109,
 111, 118, 121, 123, 128, 129, 135,
 158, 160, 185, 186, 220, 220, 221, 259
`\ifmmode` 73
`\@ifnextchar` 110, 249, 256
`\pcol@iffootnote` 83, 130, 256, 256
`\pcol@iffootnotemark` 83, 130, 256, 256
`\pcol@iffootnotetext` 110, 257, 257
`\@ifstar` 111, 242, 249, 256
`\@ifundefined` 110
`\ifvmode` 73
`\ifx` 126, 127, 242, 243, 245
`\pcol@igetcurrcol`
 121, 123, 124, 137, 185, 185
`\pcol@ignore` 108, 231, 233, 233
`\pcol@iicolumncolor` 114, 123, 260, 260, 261
`\pcol@iifootnotetext`
 87, 106, 123, 256, 257, 257
`\pcol@iigetcurrcol`
 84, 93, 94, 102, 103, 185, 185
`\pcol@iLogLevel` 125, 141, 141
`\pcol@imakeflushedpage` 83,
 85, 87, 89–93, 99–101, 105, 107, 109,
 111, 113, 119, 121, 122, 128, 129,
 133, 135, 139, 144, 158, 160, 166,
 171, 185, 186, 208, 217, 218, 218, 259
`implicit synchronization` 64
`\include` 124

infinite extension of background painting region 28, 29, 56, 77, 78, 217, 225

`\if@inlabel` 103, 228

`\ifpcol@inner` 72, 102, 134, 140, 230

`\insert` . 64–66, 68, 69, 71–74, 80–82, 88, 93, 96, 98, 101, 103, 112, 114, 121, 122, 131, 133, 138, 140–143, 146, 148, 152–156, 164, 178, 180, 183, 184, 186, 189, 191–193, 205, 206, 212, 223, 225, 231, 254, 257, 260–264

`\insertpenalties` 82

`\interfootnotelinepenalty` 82, 82

`\interlinepenalty` 69, 82, 82, 106, 178, 183, 193, 254, 255

`\pcol@invokeoutput` 81, 83, 84, 87–92, 99, 101, 128, 129, 131, 132, 137, 143, 173, 177, 226, 226, 232, 248, 249, 252, 253, 268

`\pcol@ioutputelt` 82, 85–92, 95, 96, 99, 101, 105, 109, 111, 112, 118, 121, 122, 129, 131, 134, 135, 139, 155, 156, 158, 160, 166, 171, 192, 193, 216, 219, 259

`\pcol@iremctrelt` 113, 120, 126, 127, 242, 242

`\pcol@iscancst` 83, 85–87, 91, 101, 104, 112, 114, 119, 121, 125, 127, 139, 142, 189, 189, 191

`\pcol@isetcolumnwidth` 234, 234

`\pcol@iswitchcolumn` 110, 131, 246, 246, 247

`\item` 15, 84, 98, 103, 106, 115, 228

`itemize` (environment) 15

`\itemsep` 98, 106, 228

J

`\jobname` 142

K

`\@kludgeins` 205

`\pcol@kw@fil` 240, 240

`\pcol@kw@minus` 240, 240

`\pcol@kw@plus` 240, 240

`\pcol@kw@pt` 240, 240

L

`\label` 121

`\@largefloatcheck` 115

last page 15, 51, 72, 75, 77, 81, 88, 89, 93, 95, 98, 100, 101, 103–105, 112, 116, 123, 124, 132, 133, 134, 136–139, 143, 145, 147, 157–159, 161, 171, 199, 200, 202–205, 207, 208, 210, 215–220, 222–226, 268

`\lastbox` 174, 189

`\pcol@lastcol` 225, 268, 268

`\ifpcol@lastpage` 132, 147, 179, 204, 206, 207, 215–217, 222, 226, 229, 253, 268

`\ifpcol@lastpagesave` 132, 133, 204, 206, 253

`\lastpenalty` 127, 187

`\lastskip` 106

`\@latex@warning@no@line` 113

leading column 15, 15, 17, 29, 64, 164

leading column-page 130, 203, 204

`\leftcolumn` 233, 249, 249

`leftcolumn` (environment) 8, 9, 16, 18, 18, 249, 249

`\leftcolumn*` 249

`leftcolumn*` (environment) 8, 10, 18, 18, 19, 109, 249

`\leftmargin` 15

`\let` 68, 79, 92, 94, 100, 102, 105–107, 110, 114–118, 124–127, 129, 140–143, 146, 162, 163, 165, 166, 169, 171, 179, 194, 229, 230, 233, 234, 237, 238, 242, 243, 245, 251, 252, 254, 256–258, 260, 265

`\lineskip` 106, 187

`\lineskiplimit` 106

`\linewidth` 20, 89, 89, 90, 137, 227, 229, 247

`\list` 227

`list` (environment) 15, 16, 20, 84, 89, 92, 98, 103, 106, 115, 227–229, 247

`\pcol@loadctrelt` 109, 120, 228, 243, 243, 244

local command 120, 233

local counter 10, 10, 11, 15, 15, 16, 23, 24, 66, 68, 68, 105, 113, 228, 229, 242, 243, 245, 248, 268

local representation 68, 68, 106, 228, 243, 245

`\pcol@localcommands` . . . 120, 232, 233, 233

`\localcounter` 23, 24, 24, 85, 242, 242

`\pcol@Log` 141, 141

`\pcol@Log@i` 141, 141

`\pcol@Log@ii` 128, 130, 141, 141

`\pcol@Log@iii` 128, 130, 141, 141

`\pcol@Logend` 141, 141

`\pcol@Logend@i` 141, 141

`\pcol@Logend@ii` 141, 141

`\pcol@Logfn` 141, 141

`\pcol@Logfn@i` 141, 141

`\pcol@Logfn@ii` 141, 141

`\pcol@LogLevel` 141, 141

`\pcol@Logstart` 141, 141

`\pcol@Logstart@i` 141, 141

`\pcol@Logstart@ii` 141, 141

`\long` 247, 254

`\@lowpenalty` 84

`\pcol@lrmargin` 89, 90, 137, 137, 166, 227, 229

M

<code>\@M</code>	83	<code>\pcol@marginparthreshold</code>	87, 259, 259
<code>\@m</code>	83	<code>\marginparwidth</code>	91, 194, 195
$M_{\{L,R\}}^{\{l,r\}}$	67, 67, 85, 94, 113, 119, 121, 122, 125, 127, 176, 195, 197–200, 223	<code>\mathchardef</code>	83, 84
\mathcal{M}	95, 161, 176, 198, 199–201, 222–224	<code>\maxdeadcycles</code>	226
\mathcal{M}_0	161, 198, 199, 200, 223	<code>\maxdepth</code>	28, 77, 78, 87, 92, 143, 147, 148, 154, 194
<code>\m@ne</code>	83	<code>\@maxdepth</code>	87, 88, 92, 92, 94, 118, 143–145, 147, 148, 157–159, 170, 193, 205, 214, 219, 220
<code>\pcol@magicpenalty</code>	187, 187	<code>\maxdimen</code>	87, 88, 88, 89, 106, 137, 140, 151, 152, 194, 210, 239
<code>\@makecol</code>	69, 82, 93, 99–101, 116, 116, 117, 124, 133, 136, 143– 147, 175, 176, 179, 193, 205, 210, 217	<code>\mbox</code>	60
<code>\pcol@makecol</code>	66, 69, 80, 81, 86, 88, 91, 93–96, 99, 101, 108, 109, 112, 116, 124, 128, 130, 132, 133, 137, 138, 143, 145, 145, 147, 148, 150– 152, 174, 182, 187, 192, 205, 210–213	<code>\pcol@mcid</code>	73, 130, 130, 143, 262
<code>\pcol@makecol</code>	88, 92, 100, 116, 144, 144, 147, 205, 219, 220	<code>\pcol@mcpushlimit</code>	73, 131, 262, 262
<code>\pcol@makefcolelt</code>	89, 93, 95–98, 113, 119, 123, 124, 207, 208, 208	<code>\meaning</code>	125, 127, 163
<code>\pcol@makefcolpage</code>	98, 99, 122, 124, 205, 208, 208, 219	<code>\pcol@measurecolumn</code>	83, 87, 88, 91, 93–98, 104, 120, 121, 124, 128, 136, 137, 185–187, 201, 202, 209, 209–212
<code>\@makefcolumn</code>	79, 100, 103, 118, 206, 207, 214, 215, 221	<code>\pcol@measureupdate</code>	104, 209, 210, 211, 211
<code>\pcol@makefcolumn</code>	89, 92– 95, 97, 98, 101, 104, 119–121, 123, 124, 126, 133, 205, 206, 207, 207, 208	<code>\medskip</code>	55
<code>\pcol@makeflushedpage</code>	69, 83, 85, 89, 91–93, 95, 99–101, 104, 105, 111, 113, 116, 119, 121, 122, 124, 125, 128, 129, 131–134, 136, 137, 139, 151, 155, 158, 166, 171, 180, 185, 192, 193, 202, 207, 210, 214, 215, 215, 218, 222, 223, 225	merged footnote	15, 16, 24, 24, 27, 48, 51, 55, 57, 69, 81, 82, 100, 101, 105, 133, 134, 137, 138, 176, 178, 179, 216, 218, 224, 253, 268
<code>\@makefntext</code>	116	<code>\mergedfootnotes</code>	25, 231, 233, 253, 254
<code>\pcol@makenormalcol</code>	81–83, 99–101, 104, 116, 117, 119–122, 124, 133, 134, 147, 176, 177, 178, 179, 193	<code>\message</code>	140, 141
<code>\maketitle</code>	84, 229, 247	<code>\ifpcol@mgfnote</code>	134, 134, 179, 216, 253
<code>\@marbox</code>	122, 257	<code>\@midlist</code> ..	65, 119, 121, 122, 124, 179, 205
<code>\pcol@marbox</code>	195, 195	<code>\@Mii</code>	84
<code>\marginfont</code>	23, 257	minipage (environment)	116
<code>\marginnote</code>	23, 23, 106, 108, 116, 195, 231, 257, 258	<code>\@minus</code>	109
<code>\pcol@marginnote</code>	231, 257, 257, 258	mirrored background painting	21, 28, 52, 55, 78, 78, 90, 91, 135, 136, 167, 168, 219, 221
<code>\marginpar</code>	23, 84, 85, 94, 106, 116, 194, 231, 257, 257	<code>\@Miv</code>	84
<code>\pcol@marginpar</code>	91, 106, 231, 257, 257	<code>\@MM</code>	84
<code>\pcol@marginpar</code>	106, 231, 257, 257	<code>\@mn@marginnote</code>	231, 257, 257
<code>\marginparpush</code>	91, 195	<code>\pcol@mn@warning</code>	108, 257, 258
<code>\marginparsep</code>	91, 194, 195	<code>\moveright</code>	168
<code>\marginparthreshold</code> ..	21, 22, 23, 38, 41, 44, 45, 75, 87, 110, 194, 259, 259, 270	<code>mpar(t_i, b_i)</code>	67, 67, 113, 119, 122, 127, 176, 197–201, 223
		<code>\@mparbottom</code>	67, 76, 94, 119, 151, 176, 195–197, 199, 223, 224
		<code>\pcol@mparbottom</code> ..	67, 150, 151, 194, 197
		<code>\pcol@mparbottom@out</code> ..	161, 176, 198, 198, 222
		<code>\pcol@mparbottom@zero</code> ..	119, 161, 198, 198
		<code>\pcol@mparoffset</code>	257, 257
		<code>\ifmparswitch</code>	103, 176, 199
		<code>\pcol@mpblist</code>	195, 197, 198
		<code>\pcol@mpthreshold@l</code>	194, 259, 259
		<code>\pcol@mpthreshold@r</code>	194, 259, 259
		<code>\multicolumnfootnotes</code> ..	25, 231, 233, 253, 254
		MVL-float	81, 84, 96, 98, 112, 113, 124, 148, 209, 210, 212, 212, 213

N

n_f 85, 130, 130, 134, 231, 256, 268
n_{pop} 71, 71, 73, 83, 87, 189, 190
\@namedef 109
\@nameuse 109
\@nbitem 103, 115, 228
\pcol@ncol 64, 91, 129, 129
\pcol@ncolleft 91, 129, 129
\@ne 82, 143
\newcounter 15, 124, 228
\if@newlist 103, 103, 228
\newpage 12, 107, 147, 203, 248, 249
\@next 112, 122, 142, 148, 152,
153, 157, 176, 180, 192, 206, 212, 223
\pcol@nextcol 86, 128, 129,
129, 180, 183, 203, 232, 246–248, 253
\pcol@nextpage
... 82, 86, 104, 118, 130, 149, 150, 150
\pcol@nextpelt
... 82, 83, 86, 91, 104, 118, 130, 150, 150
\pcol@tfootnotes ... 85, 130, 130, 231, 256
\@nil 85, 119, 122, 122, 123, 126,
127, 200, 201, 237, 240, 241, 256, 257
\nobackgroundcolor
... 28, 28, 56, 105, 123, 265, 265, 267
\nobreak 76, 103, 106, 183, 262, 263
\if@nobreak 65, 82, 83, 86, 103,
103, 104, 106, 115, 125, 132, 178,
181–183, 185, 186, 225, 228, 261–263
\@nobreakfalse 125, 185
\@nobreaktrue 125, 185
\nocoloredwordhyphenated . 26, 26, 264, 264
\nofncounteradjustment
..... 24, 25, 25, 134, 253, 255, 255
\nointerlineskip 105, 156, 158, 163, 166, 202
non-paired parallel-paging 16,
16, 22, 29, 42, 44–46, 67, 68, 74, 75,
85, 135, 148, 149, 152, 154, 156, 161,
163, 176, 195, 221, 225, 227, 228, 252
\normalcolor
... 25, 26, 107, 107, 114, 260, 261, 264
\normalcolseprulecolor
... 26, 26, 27, 107, 108, 110, 264, 264
\normalcolumncolor 25, 25, 71,
72, 107, 108, 110, 128, 190, 260, 260, 262
\normalfont 114
\ifpcol@nospan 131, 150, 151, 154, 215, 216
\noswapcolumninevenpages
..... 22, 135, 231, 233, 258, 258
\nthcolumn 233, 249, 249
nthcolumn (environment)
..... 8, 8, 16, 17, 17, 18, 249, 249
\nthcolumn* 249

nthcolumn* (environment)
..... 8, 17, 17, 18, 109, 249
\number 145, 180, 185, 186

O

\oddsidemargin
..... 38, 77, 89, 89, 92, 102, 162, 167
\offinterlineskip 106, 166
\pcol@op@clear 173, 227, 252
\pcol@op@end 173, 227, 268
\pcol@op@flush 173, 227, 252
\pcol@op@start 173, 227, 232
\pcol@op@switch ... 173, 227, 248, 249, 253
\@opcol 116, 143, 151, 160, 226
\pcol@opcol 71, 81, 82, 88, 91, 92, 100, 104,
112, 121, 128, 130, 132, 139, 142,
143, 148, 148–152, 154, 156, 188, 222
\@outerparskip 115
\output 70, 71, 73–76, 79–85,
88, 99, 100, 102, 102, 114–118, 128,
130, 131, 133, 134, 137, 138, 140,
142, 143, 147, 160, 166, 174, 177,
180, 182–185, 188, 194, 204, 214,
217, 222, 223, 226, 227, 229, 232,
247, 248, 252, 253, 257, 261–263, 268
\ifpcol@output 72, 131,
131, 143, 160, 161, 174, 175, 223, 263
\pcol@output .. 81–84, 87, 88, 90, 92, 93,
97, 102, 103, 105, 106, 108, 111, 113,
114, 116, 117, 131–133, 143, 143,
145, 147, 148, 164, 174, 175, 194, 232
\pcol@output@clear 79,
88, 90, 92, 93, 101, 103, 111, 116,
118, 123, 124, 129, 139, 160, 166,
170, 171, 174, 214, 214, 215, 220, 221
\pcol@output@end 67, 69, 80, 81,
83, 85, 89–95, 97, 100, 101, 103, 105,
111–113, 116, 117, 119, 122–125,
128, 129, 131, 133–140, 142, 151,
152, 158, 160, 161, 166, 170, 171,
174, 185, 189, 191, 198–200, 202,
210, 215, 216, 218–220, 222, 222, 268
\pcol@output@flush
..... 88, 90, 92, 93, 101, 129,
139, 160, 174, 214, 214, 215, 221, 222
\pcol@output@start
..... 67, 68, 80–82, 84, 91–95,
97–100, 103, 106, 108, 110–112, 116,
119, 121–125, 128–131, 134, 135,
138, 139, 142, 143, 150–152, 160,
166, 170, 171, 174, 174, 177, 179,
185, 186, 191, 199, 200, 216, 232, 262
\pcol@output@switch 66, 68,
69, 71, 81, 82, 86, 91, 93–96, 99–104,

111–113, 121, 124, 125, 127–133,
137–139, 142, 149–151, 164, 174,
177, 180, 180, 182, 183, 185–188,
192, 201, 210, 215, 247–249, 253, 268

`\@outputbox` 75, 95, 99, 100, 100, 104, 107, 116,
117, 129, 139, 144, 145, 147, 148,
152, 154–156, 158, 163, 175, 176,
179, 193, 205, 206, 214–217, 219–226

`\pcol@outputcolumns` 104,
118, 126, 132, 149, 154, 154, 179, 201

`\@outputdblcol` 160

`\pcol@outputelt` 69,
82, 85, 90, 91, 95, 99, 100, 104, 112,
118, 122, 129, 130, 132, 134, 139,
151, 154, 154, 156, 160, 166, 171, 217

`\ifpcol@outputflt` 132, 154

`\@outputpage` 82, 85, 86,
89–92, 95, 100–102, 104, 114, 117,
117, 125, 129, 131, 134–136, 139,
156, 160, 160, 162, 163, 165, 166,
171, 198, 199, 214–216, 218, 220, 224

`\pcol@@outputpage` 85,
92, 100, 102, 117, 160, 160, 162, 163

`\pcol@outputpage@ev` 92,
102, 105, 125, 127, 139, 162, 162, 163

`\pcol@outputpage@l` 85, 86,
89, 92, 102, 104, 117, 139, 161, 162, 162

`\pcol@outputpage@r` 85, 86, 89,
92, 100–102, 117, 136, 161, 162, 162, 163

`\outputpenalty`
. 81, 83, 84, 88, 125, 143, 173, 174

`\pcol@ovf` 108, 113, 142, 142

P

p_b 64, 64, 66, 83, 86, 104,
130, 132, 149, 150, 154, 156, 174, 221

p_t 64, 64, 66–70, 80, 84–86,
93–96, 99, 100, 104, 105, 111–113,
120–123, 125, 130, 132–135, 142,
145, 146, 149–153, 164, 175, 180,
181, 183, 184, 199–201, 203, 204,
206–209, 211, 215–219, 221, 225, 254

`\p@` 92

`\p@footnote` 116

`\PackageError` 108, 113,
142, 232, 239, 246, 250, 251, 253, 262

`\PackageInfo` 108

`\PackageWarning` 108, 233, 255, 260

page (counter) 11, 16, 23, 24, 42,
66, 67, 84, 85, 106, 124, 149, 228, 242

page context 66,
76, 88, 150, 183, 201, 204, 206, 215, 222

`\pcol@page`
. 65, 130, 130, 142, 149, 151, 152, 254

page rim 138

page(p) 66, 67, 74, 75, 84–86, 135, 142, 148,
149, 152–154, 156, 157, 161, 163,
167, 176, 194, 195, 199, 215, 219, 259

page-wise footnote 16,
24, 24, 27, 30, 41, 45, 48, 49, 53, 59,
60, 65, 66, 68, 68, 70, 75, 78, 80, 81,
87, 90, 93, 95, 96, 99–101, 104, 105,
107, 112, 113, 119, 125, 132–134,
137–139, 143, 146, 152, 153, 156–
158, 164, 179, 180, 183, 184, 192–
194, 203, 204, 206, 207, 214, 216–
219, 222–225, 231, 248, 253–255, 268

page-wise stuff 15,
16, 16, 27–29, 42, 50, 53, 54, 56, 57,
60, 61, 66, 68, 75, 75, 78–81, 85, 90,
98, 100, 101, 107, 112, 113, 115–118,
122–125, 134, 139, 152, 153, 155,
156, 158, 175, 214, 216, 224, 226, 229

`\pagecolor` 28

`\@pagedp` 93, 202, 223, 225

`\@pageht` 93, 195, 202, 217

`\pageref` 11, 23

`\pagerim` 29, 29, 48, 77, 91, 138, 138, 166, 170

`\pcol@pages` 66, 66, 130, 149, 154

`\pagestyle` 22, 38

`\pagetotal` 71, 140

`\ifpcol@paired`
. 135, 152, 194, 219, 221, 227, 228, 252

paired parallel-paging
. 16, 41, 42, 44, 74, 75, 135, 162, 227

`\paperheight` 28, 77, 91, 170

`\paperwidth` 28, 77, 91, 170

`\par` 17, 105, 108, 143, 227, 246, 247

`\@@par` 105, 108, 246, 249

`\pcol@par` 105, 246, 246, 247, 250, 253, 268

`\paracol` 110, 129, 135, 227, 227, 232, 260

paracol (environment) 6,
6, 7, 10–12, 15, 15, 16, 16, 18–27,
29–35, 38, 39, 41, 42, 44, 48, 50,
51, 54–57, 60, 61, 64, 66, 68, 70,
72, 74, 75, 78–81, 83–87, 89, 90, 92,
94, 97, 98, 101–103, 106–108, 115–
117, 123–125, 128–132, 134, 135,
138–140, 143, 147, 148, 152, 155,
160, 161, 163, 166, 171, 174–178,
188, 193, 194, 198, 199, 205, 223–
226, 227, 227–234, 246, 247, 249,
253, 256, 260, 262, 263, 268, 269

`\pcol@paracol` 232, 232, 260

`\paragraph` 60, 115, 247

parallel-paging . . . 16, 16, 20, 22, 23, 29,
39, 41, 42, 44–47, 60, 67, 68, 74, 74,
75, 85, 86, 90, 100, 101, 117, 125,
129, 129, 135, 138, 139, 148, 149,
152, 154–156, 158, 160–163, 171,
176, 194, 195, 214–217, 219–221,
224, 225, 227–229, 234, 235, 252, 259

`\@parboxrestore` 115

`\@parmoderr` 114

`\parshape` 92, 137, 227, 247

`\parskip` 98, 98, 106, 115, 228

`\partopsep` 98

`\penalty` 69, 82, 84, 100, 103,
106, 143, 178, 182, 183, 187, 193,
226, 232, 248, 249, 252–254, 261, 268

`\pfmtname` 108, 136, 163

`\pcol@phantom` 101, 156, 158, 158, 218

`\@plus` 108

post-environment stuff 15, 15, 16, 24, 27,
30, 39, 42, 46, 51, 52, 55, 57, 69, 75,
78, 81, 90, 93, 95, 103–105, 133, 134,
137, 139, 147, 151, 161, 171, 199,
202, 207, 216, 218, 219, 223–226, 268

pre-environment stuff 15, 16,
24, 25, 27, 32, 41, 45, 48, 51, 55, 57,
59, 66, 66–68, 75, 78, 81, 82, 88, 95,
97–101, 103, 104, 112, 116, 119–122,
124, 131, 133, 134, 138, 139, 143,
147, 152, 155, 156, 161, 171, 174,
175–177, 179, 193, 199, 200, 216, 224

pre-flushing column height check 70, 100,
131–134, 137, 201, 203, 205, 207,
208, 211, 217, 219, 227, 252, 253, 268

pre-spanning-text stuff . . 94, 97, 99, 132,
138, 145, 180, 183, 186, 187, 210, 247

`\pcol@prespan` 94,
99, 100, 138, 138, 145, 180, 183, 187, 210

`\prevdepth` 65, 83, 88, 105, 137, 177, 187,
202, 211, 212, 223, 224, 226, 227, 229

`\pcol@prevdepth`
. . . 65, 88, 93, 137, 137, 177, 226, 227

`\protect` 113

`\protected@edef` 113

`\pcol@putbackmv1`
91, 93, 97, 100, 108, 121, 131, 132,
138, 139, 183, 186, 186, 187, 189, 191

`\pcol@putfootins`
. . . . 69, 107, 114, 158, 193, 193, 218

R

R_a 77, 77, 78,
94–97, 101, 118, 119, 154, 156, 158,
159, 161, 163, 166–168, 170, 172, 176

`\raggedbottom` 17, 59, 61, 145, 213

`\raggedleftmarginnote` 23, 257

`\raggedrightmarginnote` 23, 257

`\refstepcounter` 11, 23

`\@reinserts` 186

`\relax` 107, 114,
116, 118–120, 126–128, 142, 152,
157, 161, 163, 166, 179, 192, 212,
219–221, 223, 224, 230, 232, 234,
235, 237, 238, 243, 245, 248, 249,
256, 258, 260–262, 264, 265, 267, 268

`\pcol@remctrelt` 85,
109, 120, 124, 126, 228, 242, 242, 245

`\pcol@removecounter` 120, 121, 126, 242, 242

`\renewcommand` 24

`\reserved@a`
. 120, 124, 127, 142, 149, 168, 169,
188, 189, 197, 198, 200, 201, 244, 250

`\reserved@b` 119,
125, 126, 153, 154, 188, 189, 200, 201

`\reserved@c` 80, 127, 153

`\reserved@d` 127

`\reset@color` 71, 72, 110, 114,
120, 125, 127, 188, 189, 260, 262, 263

`\pcol@reset@color@mpop` 91,
101, 103, 106, 114, 140, 262, 263, 263

`\pcol@reset@color@mpop@m` . 262, 262, 263

`\pcol@reset@color@pop` 72,
91, 101, 103, 106, 114, 131, 140, 263, 263

`\reset@font` 114

`\resetbackgroundcolor` 28, 28, 120, 268, 268

`\pcol@resetbackgroundcolor` . 120, 268, 268

`\pcol@restartcolumn` 69, 81, 82, 91,
93, 95, 96, 103, 104, 106, 113, 121,
122, 128–130, 133, 138, 139, 150,
164, 182, 183, 183, 185–187, 192, 222

`\pcol@restorecolorstack`
. 128, 140, 187, 189, 189, 225

`\pcol@restorecst` 125, 127, 139, 188, 189, 189

`\pcol@restoreeveryvbox` 140, 230, 269, 269

`\if@reversemargin` . 103, 176, 195, 199, 259

`\reversemarginpar` 22, 23, 103, 195

`\rightcolumn` 233, 249, 249

rightcolumn (environment)
. 8, 10, 16, 18, 18, 249, 249

`\rightcolumn*` 249

rightcolumn* (environment) 8, 18, 18, 109, 249

`\rightmargin` 15

`\pcol@rightpage`
. 75, 99, 100, 138, 138, 154–
156, 158, 161, 163, 176, 214–220, 225

`\romannumeral` 141

`\rule` 106

S

<i>S</i>	64, 64
S_c	64, 64, 78, 86, 99, 104, 112, 113, 148, 157, 178, 201, 204–206
$s_c(p)$	64, 64, 69, 70, 86, 112, 121, 122, 205, 208
<code>\pcol@savecolorstack</code>	100, 101, 104, 139, 140, 164, 178, 187, 191, 191
<code>\pcol@savecounters</code>	244, 244, 245, 248
<code>\pcol@savectrelet</code>	85, 113, 120, 128, 244, 244
<code>\pcol@savefootins</code>	81, 100, 112, 122, 142, 146, 180, 181, 192, 192
<code>\pcol@scancst</code>	86, 87, 101, 104, 119, 127, 139, 140, 188, 189, 189–191
<code>\pcol@scancst@shadow</code>	120, 191, 260, 260, 261
<code>\ifpcol@scfnote</code>	133, 133, 253
<code>\@scolelt</code>	117, 119
<code>\@sdblcolelt</code>	80, 118, 118, 123, 153
<code>\@sect</code>	115, 115, 251
<code>\section</code>	7, 11, 59
<code>section (counter)</code>	11
<code>\set@color</code>	71– 73, 110, 114, 114, 143, 161, 168, 178, 230, 260, 261, 262, 262, 264, 265, 267
<code>\pcol@set@color</code>	110, 114, 120, 143, 168, 178, 230, 260, 261, 262, 262
<code>\pcol@set@color@push</code>	83, 91, 96, 101, 103, 106, 108, 110, 113, 114, 120, 123, 131, 134, 140, 230, 262, 262–264
<code>\setbox</code>	139
<code>\setcolumnwidth</code>	20, 20, 21, 77, 112, 234, 234–237, 270
<code>\pcol@setcolumnwidth</code>	126, 130, 229, 234, 235, 235, 236
<code>\pcol@setcolwidth@r</code>	83, 86, 87, 90, 91, 95, 97, 111, 112, 126, 130, 234, 235, 235
<code>\pcol@setcolwidth@s</code>	83, 91, 92, 94, 96, 97, 126, 130, 234, 235, 236, 236, 239
<code>\setcounter</code>	16, 18, 23, 31
<code>\pcol@setctrelet</code>	85, 106, 120, 126, 128, 228, 245, 245, 248
<code>\pcol@setcurrcol</code>	83, 84, 86, 91, 93, 94, 102, 103, 119, 121, 123, 124, 128, 137, 181, 182, 186, 186, 209, 214
<code>\pcol@setcurrcolnf</code>	99, 177, 186, 186, 206, 220–222
<code>\pcol@setcw@accumwd</code>	94, 96, 97, 236, 238, 238
<code>\pcol@setcw@c</code>	236, 236–238
<code>\pcol@setcw@calcf</code>	83, 84, 86, 87, 91, 92, 96, 97, 108, 111, 113, 126, 239, 239, 241
<code>\pcol@setcw@calc factors</code>	90, 91, 94, 97, 126, 236, 239, 239
<code>\pcol@setcw@fill</code>	92, 98, 99, 109, 121, 127, 237, 238
<code>\pcol@setcw@filunit</code>	94, 97, 236, 236–238, 241
<code>\pcol@setcw@getspec</code>	90, 98, 112, 122, 237, 237, 238
<code>\pcol@setcw@getspec@i</code>	83, 92, 96, 98, 99, 109, 112, 121, 123, 126–128, 237, 237, 241
<code>\pcol@setcw@s</code>	236, 236–238
<code>\pcol@setcw@scale</code>	236, 238, 239, 239
<code>\pcol@setcw@scan</code>	83, 86, 111, 112, 122, 126, 130, 236, 237, 237
<code>\pcol@setcw@set</code>	86, 96, 97, 237, 238, 238
<code>\pcol@setmpbelt</code>	67, 83, 86, 91, 118–120, 125, 150, 195, 197, 198, 198
<code>\pcol@setmpbelt@i</code>	102, 125, 128, 129, 198, 198
<code>\pcol@setpageno</code>	86, 118, 121, 125, 130, 149, 149, 181, 195
<code>\pcol@setpnoelt</code>	68, 82– 84, 86, 91, 118, 135, 149, 149, 150, 198
<code>\pcol@shiftspanning</code>	85, 90, 95, 99, 135, 145, 180, 182, 182
<code>\shipout</code>	81, 102, 163
<code>\pcol@shipped@c</code>	64, 148, 150
<code>\pcol@ShowBox</code>	82, 83, 88, 91, 92, 140, 140, 141, 263
<code>\showboxbreadth</code>	82, 83
<code>\showboxdepth</code>	82, 83
<code>\pcol@shrinkcolbyfn</code>	69, 93, 96, 146, 156, 164, 183, 192, 192, 204, 206, 217
<code>\singlecolumnfootnotes</code>	25, 231, 233, 253, 254
<code>\sixt@@n</code>	83
<code>SIZE(x)</code>	202, 202, 209–211
<code>size(x)</code>	202, 202, 209, 210
<code>\skip</code>	55, 65, 66, 69, 71, 78, 81, 91, 95–99, 108, 133, 137, 138, 140, 146, 158, 164, 172, 175, 183, 186, 192– 194, 201, 203, 204, 209, 218, 219, 255
<code>\space</code>	105
<code>span(H_i, h_i)</code>	66, 66, 76, 96, 112, 113, 145, 158, 159, 180
<code>\pcol@spanning</code>	66, 150, 151
<code>spanning stuff</code>	22, 66, 66, 75, 81, 90, 92, 95, 96, 99–101, 104, 107, 112, 113, 117, 119, 122, 123, 131, 133, 134, 138, 139, 152, 153, 156, 176, 177, 179, 200, 214, 216, 219, 222, 223
<code>spanning text</code>	7, 7, 8, 11, 15, 16–18, 27, 28, 41, 44, 45, 48, 52, 60, 66, 75, 76, 85–87, 89, 90, 93, 95–97, 99–104, 115, 131, 132, 135, 138, 144, 145, 157–159, 180, 182, 183, 187, 228, 232, 247, 247–250

<code>\special</code>	70–74, 114, 123, 125, 127, 139, 140, 148, 164, 178, 180, 187–190, 260–264	<code>\pcol@swapcolumn</code>	83, 85, 129, 135, 157, 195, 219, 221, <u>259</u> , 259
<code>\@specialoutput</code>	93, <i>116</i> , 125, 143, 174, 183, 194	<code>\swapcolumninevenpages</code>	22, 75, 135, 231, 233, <u>258</u> , 258
<code>\pcol@specialoutput</code>	81–84, 99–101, 116, 125, 143, <u>174</u> , 174, 180, 194, 214, 222	<code>\ifpcol@swapmarginpar</code>	135, 136, 194, 258, 259
<code>\splitmaxdepth</code>	88, 92, 99, 193, 255	<code>\pcol@switchcol</code>	84, 92, 96, 97, 107, 108, 110, 111, 120, 128, 129, 131, 132, 173, 180, 203, 227, 228, 232, 244–246, <u>247</u> , 247, 249–251, 268
<code>\splittopskip</code>	89, <i>98</i> , 193, 255	<code>\switchcolumn</code>	6, 6–8, 12, 15, <i>16</i> , 16–19, 26, 27, 30–37, 39, 41, 44, 45, 48, 59, 60, 64, 70, 126, 128, 131, 227, 233, <u>246</u> , 246, 247, 249
<code>\ifpcol@sptext</code>	<u>131</u> , 132, 180–182, 247	<code>\pcol@switchcolumn</code>	92, 105, 108, 111, 113, 129, 130, <u>246</u> , 246, 247
<code>\pcol@sptext</code>	83, 84, 86, 87, 89–92, 102, 115, 129, 131, 132, 145, 232, 246, <u>247</u> , 247, 248	<code>\pcol@switchenv</code>	108, 113, 126, <u>249</u> , 249
<code>\pcol@sptextlist</code>	66, 145, <u>150</u> , 151, 180	<code>\pcol@switchcol</code>	248
<code>\ifpcol@sptextstart</code>	131, 138, 144, 187, <u>247</u> , 248	<code>\ifpcol@sync</code>	104, <u>131</u> , 182, 187, 201, 203, 246–248, 253
<code>\pcol@sscounters</code>	120, 121, 128, 243, <u>244</u> , 244	<code>\pcol@sync</code>	83, 85, 87, 89, 91–98, 104, 111, 128, 129, 131–133, 136, 137, 151, 154, 156, 182, <u>201</u> , 201, 203, 204, 209, 211, 222, 247, 248, 250
<code>\@ssect</code>	115	<code>\syncallcounters</code>	11, 16, <i>24</i> , 233, <u>245</u> , 245
<code>\@startcolumn</code>	<i>117</i> , 143, 164, 165, 226	<code>\pcol@synccolumn</code>	81, 83, 84, 88, 89, 91, 92, 95–98, 104, 107, 109, 112, 113, 120, 121, 124, 128, 132, 137, 142, 148, 185, 186, 203, 204, <u>211</u> , 211
<code>\pcol@startcolumn</code>	69, 81, 91, 93, 95, 96, 103, 117, 122, 123, 130, 133, 138, 139, 143, <u>164</u> , 164, 165, 186, 191, 192, 222	<code>\synccounter</code>	11, 16, <i>24</i> , 233, <u>244</u> , 244
<code>\@startdblcolumn</code>	153	<code>\pcol@synccounter</code>	83, 92, 110, 111, 120, 126, 128, 129, 229, 243, <u>244</u> , 244
starting page	15, 15, <i>66</i> , 66, 93, 95, 98, 99, 105, 124, 134, 138, 147, 150, 152, <i>174</i> , 175, 176, 178, 179, 199, 216, 217, 229	<code>\pcol@syncctrelt</code>	85, 110, 120, <u>244</u> , 244
<code>\pcol@startpage</code>	66–69, 79–82, 84, 88, 90–92, 95, 97–101, 103, 106, 107, 111, 112, 116–118, 120–124, 126, 127, 130, 133, 135, 138, 142, 146, 149–151, <u>152</u> , 152, 156, 164, 175, 222, 223	synchronize	5, 5, 7, 8, 9, 12, 16–19, 41, 44, 59, 60, <i>64</i> , 64, 65, 69, 70, 76, 81, 83, 84, 88, 89, 96–98, 102, 104, 107, 109, 111–113, 116, 117, 124, 131, 132, 137, 138, 143, 145, 147, 148, 151, 175, 179, 180, 182, 187, 201–204, 209, 211–214, 219, 226, 227, 229, 247, 248, 250
<code>\stepcounter</code>	11, 23, 31, 68, <i>106</i> , 115, 124, 152, 156, 242, 245, 256, 257		
<code>\pcol@stepcounter</code>	83, 85, 92, 110, 111, 115, 120, 124, 128, 129, 228, 242, 244, <u>245</u> , 245		
<code>\pcol@storecounters</code>	<u>243</u> , 243, 244		
<code>\pcol@storectrelt</code>	110, 113, 120, 128, <u>243</u> , 243, 244		
<code>\pcol@stpclelt</code>	85, 92, 120, <u>245</u> , 245		
<code>\@stpelt</code>	<i>115</i> , 120, 245		
<code>\pcol@stpldelt</code>	85, 120, <u>245</u> , 245		
<code>\string</code>	125, 127, 163, 260		
<code>\strutbox</code>	88, <i>99</i> , 115		
<code>\subparagraph</code>	60		
<code>\subsection</code>	29, 41, 45, 55, 59		
subsection (counter)	10, 11		
<code>\subsubsection</code>	59		
<code>\@svsec</code>	<i>115</i> , 247		
<code>\@svsechd</code>	<i>115</i> , 115, 247		
<code>\ifpcol@swapcolumn</code>	<u>135</u> , 135, 136, 161, 167, 182, 195, 228, 258, 259		

T

<code>T</code>	113, 120, 248, <i>251</i> , 251
<code>table</code> (counter)	10, 79
<code>table</code> (environment)	9, 11, 57, 59
<code>table*</code> (environment)	9, 57
<code>\@tempboxa</code>	81, <i>101</i> , 117, 153, 157, 158, 161, 166, 168, 179, 187, 189, 190, 193, 219, 221, 255
<code>\pcol@tempboxa</code>	<u>139</u> , 139, 157, 158, 160, 161, 189, 190, 220, 221
<code>\pcol@tempboxb</code>	<u>139</u> , 139, 189

`\@tempcnta` 83,
 85, 118, 157, 247, 250, 256, 257, 259
`\@tempcntb` 87, 157, 189, 202
`\@tempdima` 95, 154, 156,
 168, 192, 195, 202, 227, 238, 250, 268
`\@tempdimb` . 96, 158, 159, 164, 168, 183,
 192, 193, 195, 202, 203, 236–239, 241
`\@tempdimc` 97, 168, 202
`\@tempskipa` 83, 92, 99, 109, 238, 240
`\if@tempswa` 103, 150, 154, 189, 190, 203,
 209–211, 216, 217, 221, 223, 252, 265
`\@tempswafalse` 105, 251
`\@tempswatru` 105, 251, 252
`\@temptokena` 102
`\@testwrongwidth` 79
`\@textbottom` 108,
 109, 116, 145, 147, 148, 179, 216, 217
`\pcol@textbottom` 116, 145
`\textcolor` 26, 73
`\textfloatsep` 15, 65, 89,
 96, 98, 117, 133, 137, 147, 148, 175,
 176, 179, 201, 202, 209, 210, 212, 213
`\pcol@textfloatsep`
 65, 84, 88, 96, 137, 137,
 148, 151, 175, 179, 209, 210, 226, 229
`\@textfloatsheight` 65, 94, 151
`\textheight`
 . 24, 28, 66, 78, 90, 92, 93, 97, 108,
 151–154, 156, 161, 170, 176, 179,
 193, 194, 214, 215, 220, 221, 226, 255
`\textwidth` 19–21, 77, 86,
 87, 89, 90, 90, 95, 97, 107, 109, 134,
 137, 145, 157, 182, 195, 219, 221,
 229, 231, 235, 236, 239, 247, 255, 268
`\@tfor` 112, 126, 127, 237
`\the` 102, 230, 238–240
`\the θ` 68, 106, 126, 228, 243, 245
`\thecolumn` 18, 18, 128, 233, 233, 270
`\pcol@thectr@ θ` 106, 126, 228, 243
`\pcol@thectr@ θ -c`
 . . 68, 106, 109, 126, 228, 243, 243, 245
`\pcol@thectrelt` 106, 120, 126, 228, 243, 243
`\pcol@thecurrcol` 243
`\@thefnmark` 116, 121
`\thefootnote` 12, 116
`\theH θ` 243
`\@themargin` 92, 162, 163
`\thepage` 106
`\thesection` 115
`\thesubsection` 11, 12
`\toks` 102, 140
top page 29,
 29, 64, 67–69, 99, 101, 130, 133,
 145, 153, 206, 207, 214, 222, 248, 253
`\topfigrule` 107, 148, 210, 212, 219
`\pcol@topfnotes` 69, 138, 138, 225, 253, 254
`\topfraction` 177
`\@toplist` 65, 101,
 113, 117, 119, 124, 145, 165, 205–208
`\topmargin` 77,
 89, 89, 90, 95, 96, 156, 160, 170, 216
`\@topnum` 65, 84, 84
`\pcol@toppage` 64, 130, 130, 142
`\@toproom` 65, 93, 219
`\topsep` 98
`\@topsep` 98, 106, 228
`\topskip` 66, 97, 99, 138, 150–152,
 175–177, 187, 212, 213, 224, 229, 268
`\pcol@topskip` 66,
 97, 138, 138, 151–153, 175, 229, 268
`\@totalleftmargin` 92, 227, 247
trivlist (environment) 103, 228
`\@trivlist` 98
true 65, 72, 73,
 77, 82, 83, 102–106, 111, 114, 115,
 124, 131–136, 138, 140, 143–145,
 147, 150–152, 154, 155, 160, 161,
 166–168, 174, 176, 178–183, 187,
 189, 194–196, 198, 199, 201–204,
 206, 207, 209–211, 215–217, 222–
 224, 226–230, 246–248, 252, 253,
 255, 258, 259, 261–263, 265, 268, 297
`\@tryfcolumn` 79, 100, 103, 117, 148, 152, 164
`\pcol@trynextcolumn` 117, 119, 121,
 123, 124, 126, 147, 164, 165, 165, 206
`\tw@` 83
`\twocolumn` 84, 229, 247
`\if@twocolumn`
 102, 143, 195, 215, 227, 229, 268
`\if@twoside` 102, 167, 252, 258
`\twosided` 21, 22, 23, 28, 29,
 38, 38, 42, 44, 48, 48, 52, 55, 75, 77,
 78, 102, 110, 135, 194, 231, 233, 258, 258
`\pcol@twosided` 102, 108,
 110, 112, 126, 135, 231, 233, 258, 258
`\pcol@twosided@t` 110
`\pcol@twosided@b` 110, 135, 258
`\pcol@twosided@c` 110, 135, 258
`\pcol@twosided@m` 110, 135, 258
`\pcol@twosided@p` 102, 110, 258

U

`\unskip` 193
`\unvbox` 125, 148, 156, 177–179, 189, 193, 224
`\pcol@unvbox@cclv`
 69, 92, 95, 99, 146, 192, 192, 205
`\unvcopy` 127, 145, 187, 189
`\usepackage` 6

<code>\output</code>	122	W_P	77, 77, 78, 91, 167, 169, 170, 172
V			
V_B	89, 92, 94, 96, 202, 202, 203, 209	W_R	77, 77, 78, 91, 169, 170, 172
V_E	94, 96, 132, 203, 204, 248, 250	W_T	77, 77, 78, 90, 94, 97, 126, 145, 160, 161, 167, 168, 172, 182, 195, 219, 236–239
V_P	87, 89, 93, 94, 96, 104, 133, 202, 202, 203, 209	w_c	19, 20, 66, 77, 86, 87, 89, 90, 95–97, 111, 137, 157, 160, 185, 195, 221, 227, 229, 234–236, 239
V'_P	89, 95, 98, 104, 105, 133, 137, 202, 202, 203, 210, 211, 215–218, 222	<code>\whiledim</code>	111
V_T	89, 92, 94–96, 104, 202, 202, 203, 209, 211–213	<code>\whilenum</code>	83, 111
$v_c(x)$	96, 201, 201, 202, 212, 213	<code>\whilesw</code>	111
$val(\theta)$	85, 86, 105, 229, 229, 243–245, 248	<code>\width</code>	108
$val_c(\theta)$	68, 68, 85, 86, 105, 109, 113, 126, 228, 229, 242–245, 248	X	
<code>\vbadness</code>	82, 83, 179, 193, 194	<code>\pcol@xcolumncolor</code>	107, 260, 260
<code>\vbox</code>	64, 65, 69–73, 82, 88, 96, 99, 101, 102, 105, 114, 115, 127, 134, 137, 140, 148, 156, 163, 165, 189, 191, 201, 208, 213, 216, 219, 221, 222, 226, 230, 231, 254, 255, 261–263	<code>\xdef</code>	73, 118–120, 150, 176, 186, 191, 199, 238, 244, 261, 267, 268
<code>verse</code> (environment)	19	<code>\xfloat</code>	81, 84
<code>\vfil</code>	116, 161, 204, 205, 211, 216, 217, 219, 248, 253	<code>\@xnext</code>	112, 128, 195
<code>\vfuze</code>	88, 140	<code>\pcol@xparacol</code>	227, 227
<code>\pcol@visitallcols</code>	83, 87, 92, 107, 108, 111, 128, 129, 173, 180, 227, 248, 248, 249, 253	<code>\@xympar</code>	116, 231, 257, 257
<code>\voidb@x</code>	99, 146, 157, 177, 186, 191	<code>\pcol@xympar</code>	84, 92, 116, 122, 231, 257
<code>\vrule</code>	108, 168	<code>\pcol@xympar</code>	116, 231, 257
<code>\vsize</code>	82, 87, 88, 93, 143, 144, 174	Y	
<code>\vskip</code>	79, 140, 158, 166	<code>\pcol@ycolumncolor</code>	107, 260, 260
<code>\vsplit</code>	82, 98, 193	<code>\yoko</code>	125, 127, 163
<code>\vss</code>	212, 255	<code>\pcol@yparacol</code>	111, 135, 227, 227, 228
<code>\vtop</code>	166, 168	Z	
W			
W_c	77, 77, 78, 170, 172, 239	<code>\z@</code>	79, 91, 124, 257, 259, 262, 264
W_M	77, 77, 78, 167, 169, 170, 172	<code>\pcol@zparacol</code>	79, 83–85, 88–91, 97, 98, 102, 103, 105–110, 113–117, 120, 121, 124, 128–130, 133–135, 137, 138, 140, 143, 146, 147, 161, 166, 173–175, 177–179, 194, 195, 227, 227, 233–236, 242–244, 247, 249, 254, 256, 257, 262, 269

Revision History

v0.9	
General: The style paracol is born. (2005/01/28)	1
v0.91	
General: The style is included in CTAN with a very small modification. (2011/09/16)	1
v1.0	
General: Add this document and fix the following problems on the author's 30th wedding anniversary. (2011/10/10)	1
<code>\pcol@toppage</code> : Renamed from <code>\pcol@maxpage</code> .	130
<code>\ifpcol@nospan</code> : Renamed from <code>\pcol@textonly</code> .	131
<code>\ifpcol@sync</code> : Add initialization to be <i>false</i> .	131
<code>\ifpcol@spstext</code> : Introduced to restrict the broadcast of <code>\if@nobreak</code> and <code>\everypar</code> only when a column-switching is accompanied with spanning text.	131
<code>\ifpcol@clear</code> : Add initialization to be <i>false</i> .	132
<code>\ifpcol@outputflt</code> : Renamed from <code>\ifpcol@stopoutput</code> with the reversal.	132
<code>\ifpcol@lastpage</code> : Introduced for special operations in the last page.	132
<code>\pcol@textfloatsep</code> : Introduced for the bug fix of float space enlargement.	137
<code>\pcol@output</code> : Replace <code>\@makecol</code> with <code>\pcol@makecol</code> for a special care for column-pages with synchronization points.	143
<code>\pcol@makecol</code> : Introduced for special float handling in a column-page with synchronization points.	145
<code>\pcol@combinefloats</code> : Introduced for special float handling in a column-page with synchronization points.	146
<code>\pcol@cflt</code> : Introduced for special float handling in a column-page with synchronization points.	148
<code>\pcol@opcol</code> : Remove unnecessary assignment of <code>\@colht</code> .	148
<code>\pcol@opcol</code> : Rename <code>\pcol@maxpage</code> as <code>\pcol@toppage</code> .	148
<code>\pcol@setpnoelt</code> : Rename <code>\ifpcol@textonly</code> as <code>\ifpcol@nospan</code> .	149
<code>\pcol@getpinfo</code> : Rename <code>\ifpcol@textonly</code> as <code>\ifpcol@nospan</code> .	150
<code>\pcol@floatplacement</code> : Add initialization of <code>\pcol@textfloatsep</code> .	151
<code>\pcol@startpage</code> : Add assignment of <code>\pcol@firstprevdepth</code> to be <code>\relax</code> .	152
<code>\pcol@startpage</code> : Rename <code>\pcol@maxpage</code> as <code>\pcol@toppage</code> .	152
<code>\pcol@startpage</code> : Rename <code>\pcol@setttextpage</code> as <code>\pcol@setordpage</code> .	152
<code>\pcol@outputcolumns</code> : Rename <code>\ifpcol@stopoutput</code> as <code>\ifpcol@outputflt</code> .	154
<code>\pcol@outputelt</code> : Rename <code>\ifpcol@stopoutput</code> as <code>\ifpcol@outputflt</code> .	155
<code>\pcol@outputelt</code> : Rename <code>\ifpcol@textonly</code> as <code>\ifpcol@nospan</code> .	155
<code>\pcol@specialoutput</code> : Remove unnecessary <code>\pcol@latex@specialoutput</code> .	174
<code>\pcol@output@start</code> : Change the order of operations.	174
<code>\pcol@output@start</code> : Rename <code>\pcol@maxpage</code> as <code>\pcol@toppage</code> .	174
<code>\pcol@output@start</code> : Add special operation in case of too small room for column-pages.	174
<code>\pcol@output@start</code> : Add clearing of S_c .	178
<code>\pcol@output@switch</code> : Restrict the broadcast of <code>\if@nobreak</code> and <code>\everypar</code> only when a column-switching is accompanied with spanning text.	182
<code>\pcol@restartcolumn</code> : Add <code>\pcol@getcurrfoot</code> to restore parameters of $\kappa_c(\tau)$ into <code>\footins</code> .	183
<code>\pcol@iigetcurrcol</code> : Add restoration of <code>\pcol@textfloatsep</code> .	185
<code>\pcol@setcurrcol</code> : Add save of <code>\pcol@textfloatsep</code> .	186
<code>\pcol@sync</code> : Add measurement of D_T .	201
<code>\pcol@flushcolumn</code> : Rename <code>\pcol@maxpage</code> as <code>\pcol@toppage</code> .	204
<code>\pcol@flushcolumn</code> : Add <code>\vfil</code> at the bottom of flushed column-page.	204
<code>\pcol@flushcolumn</code> : Change order of the garbage collection of <code>\pcol@currfoot</code> and <code>\pcol@getcurrfoot</code> .	204
<code>\pcol@flushcolumn</code> : Add <code>\@colht = \pi^h(p)</code> .	206

<code>\pcol@flushcolumn</code> : Replace <code>\pcol@trynextcolumn</code> with <code>\pcol@makefcolumn</code> for the case of <code>\ifpcol@clear = true</code>	206
<code>\pcol@makefcolumn</code> : Introduced to take special care of the float-column in the last page.	207
<code>\pcol@makefcolelt</code> : Introduced to take special care of the float-column in the last page.	208
<code>\pcol@measurecolumn</code> : Drastically changed to measure D_T , to deal with empty main vertical list, and to omit <code>\topfigrule</code> and <code>\botfigrule</code> from float size measurement.	209
<code>\pcol@addflhd</code> : Drastically changed to omit <code>\topfigrule</code> and <code>\botfigrule</code> from float size measurement, to take care of top float enlargement, to add the measurement of D_T , and to revise the definition of D_P	210
<code>\pcol@measureupdate</code> : Introduced to let D_T and D_P have the minimum depth of items among those which give V_T and V_P	211
<code>\pcol@synccolumn</code> : Drastically changed to correctly implement the top float enlargement and MVL-float.	211
<code>\pcol@output@flush</code> : Rename <code>\pcol@makelastpage</code> as <code>\pcol@makeflushedpage</code>	214
<code>\pcol@output@flush</code> : Remove unnecessary assignment of <code>\@colht</code>	214
<code>\pcol@output@clear</code> : Rename <code>\pcol@makelastpage</code> as <code>\pcol@makeflushedpage</code>	214
<code>\pcol@output@clear</code> : Remove unnecessary increment of <code>\pcol@page</code> and assignment of <code>\@colht</code>	214
<code>\pcol@makeflushedpage</code> : Renamed from <code>\pcol@makelastpage</code>	215
<code>\pcol@makeflushedpage</code> : Rename <code>\ifpcol@textonly</code> as <code>\ifpcol@nospan</code>	215
<code>\pcol@makeflushedpage</code> : Judge the last page is empty if $V_P = -\infty$ instead of $V_P < 0$	215
<code>\pcol@makeflushedpage</code> : Let <code>\@colht</code> be $\langle ht \rangle$ if the former is less than the latter.	217
<code>\pcol@makeflushedpage</code> : Add special care of the float column in the last page.	217
<code>\pcol@flushfloats</code> : Add reinitialization of <code>\@colht</code>	220
<code>\pcol@freshpage</code> : Rename <code>\pcol@maxpage</code> as <code>\pcol@toppage</code>	221
<code>\pcol@freshpage</code> : Add save and restore of <code>\@currbox</code>	221
<code>\pcol@freshpage</code> : Remove unnecessary assignment of <code>\pcol@currcol</code>	221
<code>\pcol@output@end</code> : Drastically changed to take special care of float columns in the last page, to deal with the empty last page with and without deferred floats, and to try to make post-environment float pages.	222
<code>\pcol@invokeoutput</code> : Let <code>\pcol@prevdepth</code> have <code>\prevdepth</code> directly instead of via <code>\tempdima</code>	226
<code>\paracol</code> : Change the order of operations for sake of clarity.	227
<code>\paracol</code> : Add the mechanism of inter-environment local counter conservation.	227
<code>\paracol</code> : Let <code>\col@number = 1</code> instead of C to keep <code>\maketitle</code> from producing title with <code>\twocolumn</code>	227
<code>\paracol</code> : Add initialization of <code>\pcol@textfloatsep</code> , <code>\ifpcol@lastpage</code> , <code>\pcol@firstprevdepth</code> and <code>\@combinefloats</code>	227
<code>\paracol</code> : Make API commands environment-local and inhibit nesting of <code>paracol</code>	227
<code>\pcol@localcommands</code> : Introduced to make API commands environment-local.	233
<code>\pcol@defcomelt</code> : Introduced to make API commands environment-local.	233
<code>\pcol@loadctrelt</code> : Introduced for inter-environment local counter conservation.	243
<code>\pcol@storecounters</code> : Introduced for inter-environment local counter conservation.	243
<code>\pcol@storectrelt</code> : Introduced for inter-environment local counter conservation.	243
<code>\pcol@savecounters</code> : Move the body to <code>\pcol@sscounters</code>	244
<code>\pcol@sscounters</code> : Introduced to implement the common operations of <code>\pcol@storecounters</code> and <code>\pcol@savecounters</code>	244
<code>\pcol@cmpctrelt</code> : Introduced for inter-environment local counter conservation.	244
<code>\synccounter</code> : Introduced as an environment-local API command.	244
<code>\pcol@com@synccounter</code> : Introduced for the implementation of the new API command <code>\synccounter</code>	244
<code>\pcol@synccounter</code> : Introduced for <code>\synccounter</code> and inter-environment local counter conservation.	244

<code>\pcol@syncctrelt</code> : Introduced for <code>\synccounter</code> and inter-environment local counter conservation.	244
<code>\syncallcounters</code> : Introduced as an environment-local API command.	245
<code>\pcol@com@syncallcounters</code> : Introduced for the implementation of the new API command <code>\syncallcounters</code>	245
<code>\pcol@setctrelt</code> : Replace <code>\setcounter</code> with direct assignment with <code>\csname</code> and <code>\endcsname</code>	245
<code>\pcol@stepcounter</code> : Change the order of operations.	245
<code>\pcol@stepcounter</code> : Replace <code>\csname/\endcsname</code> with <code>\@nameuse</code>	245
<code>\pcol@par</code> : Introduced for <code>\par-if-necessary</code> operation.	246
<code>\switchcolumn</code> : Made <code>\let-equal</code> to <code>\pcol@com@switchcolumn</code> for localization.	246
<code>\pcol@com@switchcolumn</code> : Introduced as the implementation of <code>\switchcolumn</code>	246
<code>\pcol@com@switchcolumn</code> : Add <code>\pcol@defcolumn</code> to clarify the behavior of <code>column</code>	246
<code>\pcol@switchcolumn</code> : Add the check of $d \geq 0$	246
<code>\pcol@sptext</code> : Made <code>\long</code> to allow <code>\par</code> in its argument.	247
<code>\pcol@sptext</code> : Replace <code>\par</code> with <code>\pcol@par</code>	247
<code>\pcol@sptext</code> : Add <code>\ifpcol@cmtext = true</code> for restriction of the broadcast of <code>\if@nobreak</code> and <code>\everypar</code>	247
<code>\pcol@switchcol</code> : Add <code>\pcol@aconly</code> for disabling <code>\addcontentsline</code>	247
<code>\column</code> : Made <code>\let-equal</code> to <code>\pcol@com@column</code> for localization.	249
<code>\column*</code> : Made <code>\let-equal</code> to <code>\pcol@com@column*</code> for localization.	249
<code>\pcol@com@column</code> : Introduced as the implementation of <code>\column</code>	249
<code>\pcol@com@column*</code> : Introduced as the implementation of <code>\column*</code>	249
<code>\pcol@defcolumn</code> : Replace <code>\column</code> with <code>\pcol@com@column</code> and <code>\switchcolumn</code> with <code>\pcol@switchenv</code>	249
<code>\nthcolumn</code> : Made <code>\let-equal</code> to <code>\pcol@com@nthcolumn</code> for localization.	249
<code>\nthcolumn*</code> : Made <code>\let-equal</code> to <code>\pcol@com@nthcolumn*</code> for localization.	249
<code>\pcol@com@nthcolumn</code> : Introduced as the implementation of <code>\nthcolumn</code> with the inhibition of column-switching in the environment.	249
<code>\pcol@com@nthcolumn*</code> : Introduced as the implementation of <code>\nthcolumn*</code> with the inhibition of column-switching in the environment.	249
<code>\leftcolumn</code> : Made <code>\let-equal</code> to <code>\pcol@com@leftcolumn</code> for localization.	249
<code>\leftcolumn*</code> : Made <code>\let-equal</code> to <code>\pcol@com@leftcolumn*</code> for localization.	249
<code>\pcol@com@leftcolumn</code> : Introduced as the implementation of <code>\leftcolumn</code> with the inhibition of column-switching in the environment.	249
<code>\pcol@com@leftcolumn*</code> : Introduced as the implementation of <code>\leftcolumn*</code> with the inhibition of column-switching in the environment.	249
<code>\rightcolumn</code> : Made <code>\let-equal</code> to <code>\pcol@com@rightcolumn</code> for localization.	249
<code>\rightcolumn*</code> : Made <code>\let-equal</code> to <code>\pcol@com@rightcolumn*</code> for localization.	249
<code>\pcol@com@rightcolumn</code> : Introduced as the implementation of <code>\rightcolumn</code> with the inhibition of column-switching in the environment.	249
<code>\pcol@com@rightcolumn*</code> : Introduced as the implementation of <code>\rightcolumn*</code> with the inhibition of column-switching in the environment.	249
<code>\pcol@switchenv</code> : Introduced to inhibit column-switching in the environment.	249
<code>\endcolumn</code> : Made <code>\let-equal</code> to <code>\pcol@com@endcolumn</code> for localization.	250
<code>\endcolumn*</code> : Made <code>\let-equal</code> to <code>\pcol@com@endcolumn*</code> for localization.	250
<code>\pcol@com@endcolumn</code> : Introduced as the implementation of <code>\endcolumn</code> with the globalization of <code>\everypar</code>	250
<code>\pcol@com@endcolumn*</code> : Introduced as the implementation of <code>\endcolumn*</code> with the globalization of <code>\everypar</code>	250
<code>\endnthcolumn</code> : Made <code>\let-equal</code> to <code>\pcol@com@endnthcolumn</code> for localization.	250
<code>\endnthcolumn*</code> : Made <code>\let-equal</code> to <code>\pcol@com@endnthcolumn*</code> for localization.	250
<code>\pcol@com@endnthcolumn</code> : Introduced as the implementation of <code>\endnthcolumn</code> with the globalization of <code>\everypar</code>	250

<code>\pcol@com@endnthcolumn*</code> : Introduced as the implementation of <code>\endnthcolumn*</code> with the globalization of <code>\everypar</code>	250
<code>\endleftcolumn</code> : Made <code>\let</code> -equal to <code>\pcol@com@endleftcolumn</code> for localization.	250
<code>\endleftcolumn*</code> : Made <code>\let</code> -equal to <code>\pcol@com@endleftcolumn*</code> for localization.	250
<code>\pcol@com@endleftcolumn</code> : Introduced as the implementation of <code>\endleftcolumn</code> with the globalization of <code>\everypar</code>	250
<code>\pcol@com@endleftcolumn*</code> : Introduced as the implementation of <code>\endleftcolumn*</code> with the globalization of <code>\everypar</code>	250
<code>\endrightcolumn</code> : Made <code>\let</code> -equal to <code>\pcol@com@endrightcolumn</code> for localization.	250
<code>\endrightcolumn*</code> : Made <code>\let</code> -equal to <code>\pcol@com@endrightcolumn*</code> for localization.	250
<code>\pcol@com@endrightcolumn</code> : Introduced as the implementation of <code>\endrightcolumn</code> with the globalization of <code>\everypar</code>	250
<code>\pcol@com@endrightcolumn*</code> : Introduced as the implementation of <code>\endrightcolumn*</code> with the globalization of <code>\everypar</code>	250
<code>\addcontentsonly</code> : Introduced for disabling <code>\addcontentsline</code>	250
<code>\pcol@aonly</code> : Introduced for disabling <code>\addcontentsline</code>	250
<code>\pcol@aonlyelt</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@gobblethree</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@addcontentsline</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@ac@def@toc</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@ac@enable@toc</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@ac@disable@toc</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@ac@def@lof</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@ac@def@lot</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@ac@caption@enable</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@ac@caption@disable</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@ac@caption@def</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@ac@caption@if@lof</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@ac@caption@if@lot</code> : Introduced for disabling <code>\addcontentsline</code>	251
<code>\pcol@ac@caption</code> : Introduced for disabling <code>\addcontentsline</code>	252
<code>\pcol@ac@caption@latex</code> : Introduced for disabling <code>\addcontentsline</code>	252
<code>\flushpage</code> : Made <code>\let</code> -equal to <code>\pcol@com@flushpage</code> for localization.	252
<code>\pcol@com@flushpage</code> : Introduced as the implementation of <code>\flushpage</code> with the replacement of <code>\par</code> with <code>\pcol@par</code>	252
<code>\clearpage</code> : Made <code>\let</code> -equal to <code>\pcol@com@clearpage</code> for localization.	252
<code>\pcol@com@clearpage</code> : Introduced as the implementation of <code>\clearpage</code> with the replacement of <code>\par</code> with <code>\pcol@par</code>	252
<code>\endparacol</code> : Replace <code>\par</code> with <code>\pcol@par</code>	268
v1.1	
General: Add <code>\columnratio</code> and variable column width. (2012/05/11)	1
General: Add <code>\columnratio{0.6}</code> and a phrase for it.	12
General: Add description of <code>\columnratio</code>	19
General: Add description of $w_c = \pcol@columnwidth \cdot c$	66
<code>\pcol@outputelt</code> : Use <code>\pcol@columnwidth \cdot c</code> instead of <code>\columnwidth</code> as the width of c 's <code>\hbox</code>	155
<code>\pcol@getcurrcol</code> : Add assignment w_c to <code>\columnwidth</code> , <code>\hsize</code> and <code>\linewidth</code>	185
<code>\pcol@zparacol</code> : Replace the calculation of <code>\columnwidth</code> with the call of <code>\pcol@setcolumnwidth</code> and the assignment of w_0 to it.	229
<code>\columnratio</code> : Introduced to specify column width fractions.	234
<code>\pcol@columnratioleft</code> : Introduced to keep column width fractions.	234
<code>\pcol@setcolwidth@r</code> : Introduced to calculate w_c	235
v1.2-1	
General: Make <code>paracol</code> environment accept <code>\color</code> and add <code>\columncolor</code> . (2013/05/11) . . .	1
General: Add description of <code>\columncolor</code> and <code>\normlcolumncolor</code>	25

General: Add the subsection “Coloring”.	70
<code>\pcol@currcol</code> : Add initialization to 0 after the declaration.	128
<code>\pcol@opcol</code> : Add <code>\pcol@clearcst@unvbox</code> to add coloring <code>\specials</code> at the top and bottom of the column-page to be shipped out, together with the setting <code>\boxmaxdepth = \@maxdepth</code> for depth capping.	148
<code>\pcol@output@start</code> : Add emptying <code>\pcol@colorstack</code> and the invocation of <code>\pcol@savecolorstack</code> at the beginning of the first column-page to be built.	178
<code>\pcol@output@switch</code> : Add <code>\pcol@clearcst@unvbox</code> to add coloring <code>\specials</code> at the top and bottom of the column-page to be saved.	180
<code>\pcol@restartcolumn</code> : Add <code>\pcol@restorecst@restart</code> to return main vertical list with coloring operations.	183
<code>\pcol@putbackmv1</code> : Introduced to restart a column with coloring.	186
General: Add the subsection “Color Management” to describe newly introduced macros for coloring.	187
General: <code>\pcol@reset@color@elt</code> was introduced to implement <code>\pcol@reset@color@pop</code> but removed in v1.34.	187
<code>\pcol@ifempty</code> : Introduced to examine the emptiness of the box, extracting the code from <code>\pcol@measurecolumn</code> so as to be used for coloring as well.	187
<code>\pcol@clearcst@unvbox</code> : Introduced to put coloring <code>\specials</code> above and below of a column-page.	188
<code>\pcol@clearcolorstack</code> : Introduced to clear color context.	188
General: <code>\pcol@set@color@elt</code> was introduced to implement color context reestablishment but removed in v1.34.	188
<code>\pcol@restorecolorstack</code> : Introduced to reestablish color context.	189
<code>\pcol@restorecst</code> : Introduced to reestablish color context.	189
<code>\pcol@savecolorstack</code> : Introduced to save the opening color context of a column-page.	191
General: <code>\pcol@colorstack@full</code> was introduced to represent I^c but removed in v1.34.	191
<code>\pcol@output@end</code> : Add color resetting.	226
<code>\pcol@zparacol</code> : Add redefinitions of <code>\set@color</code> and <code>\reset@color</code> .	230
<code>\columncolor</code> : Introduced to define the default color of a column.	260
<code>\pcol@xcolumncolor</code> : Introduced to implement <code>\columncolor</code> .	260
<code>\pcol@ycolumncolor</code> : Introduced to implement <code>\columncolor</code> .	260
<code>\pcol@columncolor</code> : Introduced to implement <code>\columncolor</code> .	260
<code>\normalcolumncolor</code> : Introduced to define the default color of a column is <code>\normalcolor</code> .	260
<code>\pcol@icolumncolor</code> : Introduced to implement <code>\columncolor</code> and <code>\normalcolumncolor</code> .	260
<code>\pcol@set@color@push</code> : Introduced to work as <code>\set@color</code> .	262
<code>\pcol@reset@color@pop</code> : Introduced to work as <code>\reset@color</code> .	263
v1.2-2	
General: Add page-wise and merged footnote functions. (2013/05/11)	1
General: Add a footnote mentioning page-wise footnotes.	10
General: Add a footnote mentioning page-wise footnotes merged with pre-environment staff.	15
General: Add a footnote mentioning page-wise footnotes merged with post-environment staff.	15
General: Add the sub-section “Single-Columned Footnotes” to describe newly introduced commands for page-wise footnotes.	24
General: Add the section “Numbering and Placement of Single-Columned Footnotes” to describe page-wise footnotes in detail.	30
General: Redesign page context and its implementation.	66
General: Add the subsection “Single-Columned and Merged Footnotes”.	68
<code>\pcol@footnotebase</code> : Introduced for page-wise footnotes.	130
<code>\pcol@nfootnotes</code> : Introduced for page-wise footnotes.	130
<code>\ifpcol@sync</code> : Add <code>\pcol@switchcol</code> and <code>\pcol@flushclear</code> to the macros turning the switch <code>true</code> due to column scanning and pre-flushing column height check.	131
<code>\ifpcol@clear</code> : Add <code>\pcol@flushclear</code> to the macros turning the switch <code>true</code> due to the pre-flushing column height check.	132

<code>\ifpcol@flush</code> : Add uses for page-wise footnote functions.	132
<code>\ifpcol@scfnote</code> : Introduced for page-wise footnote functions.	133
<code>\ifpcol@mgfnote</code> : Introduced for page-wise footnote functions.	134
<code>\ifpcol@fncounteradjustment</code> : Introduced for page-wise footnote functions.	134
<code>\pcol@topfnnotes</code> : Introduced for page-wise footnote functions.	138
<code>\pcol@ShowBox</code> : Introduced for debugging page-wise footnote functions.	140
<code>\pcol@LogLevel</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@iLogLevel</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Log</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Log@iii</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Log@ii</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Log@i</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Logstart</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Logstart@ii</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Logstart@i</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Logend</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Logend@ii</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Logend@i</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Logfn</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Logfn@ii</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@Logfn@i</code> : Introduced for debugging page-wise footnote functions.	141
<code>\pcol@output</code> : Add <code>\pcol@Logstart</code> and <code>\pcol@Logend</code>	143
<code>\pcol@output</code> : Add an argument <code>\@ne</code> to <code>\pcol@startcolumn</code> to distinguish it from the invocation in <code>\pcol@freshpage</code>	143
<code>\pcol@makecol</code> : Add save/discard of page-wise footnotes.	145
<code>\pcol@combinefloats</code> : Remove the shrink of <code>\textfloatsep</code> because each column in the last page is now made not taller than <code>\@colht</code> definitely by the introduction of the pre-flushing column height check.	146
<code>\pcol@setpnoelt</code> : Completely recode reflecting the redesign of page context.	149
<code>\pcol@defcurrpage</code> : Introduced by the redesign of page context, partly replacing <code>\pcol@setordpage</code> which once we call <code>\pcol@settexpage</code>	150
<code>\pcol@nextpelt</code> : Revise reflecting the redesign of page context.	150
<code>\pcol@getpelt</code> : Revise reflecting the redesign of page context.	150
<code>\pcol@getpinfo</code> : Revise reflecting the redesign of page context.	150
<code>\pcol@getcurrpinfo</code> : Revise reflecting the redesign of page context.	150
<code>\pcol@startpage</code> : Revise reflecting the redesign of page context.	152
<code>\pcol@outputelt</code> : Revise reflecting the redesign of page context.	154
<code>\pcol@outputelt</code> : Add ship-out of page-wise footnote.	155
<code>\pcol@trynextcolumn</code> : No change in the code itself but its explanation is modified according to the drastic redesign of <code>\pcol@startcolumn</code>	165
<code>\pcol@specialoutput</code> : Add footnote logging if <code>\outputpenalty = -10004</code>	174
<code>\pcol@output@start</code> : Add initialization of Φ and page-wise footnote output operations, and revise reflecting the redesign of page context.	174
<code>\pcol@output@start</code> : Add insertion of <code>\footins</code> having footnotes to be merged if it is not void.	178
<code>\pcol@makenormalcol</code> : Replace the building operation of <code>\@outputbox</code> with footnotes with the invocation of <code>\pcol@combinefootins</code> and add the examination of <code>\ifpcol@mgfnote</code>	178
<code>\pcol@output@switch</code> : Add the case <code>\ifpcol@clear = \ifpcol@sync = true</code> for the invocation of <code>\pcol@restartcolumn</code> for pre-flushing column height check.	182
<code>\pcol@restartcolumn</code> : Redesign the footnote insertion mechanism to cope with page-wise footnotes.	183
<code>\pcol@getcurrfoot</code> : Add an argument being <code>\box</code> or <code>\copy</code>	186
General: Add the subsection “Footnote Handling” to describe newly introduced macros for page-wise footnotes.	192

<code>\pcol@savefootins</code> : Introduced to save footnotes in multiple occasions.	192
<code>\pcol@shrinkcolbyfn</code> : Introduced to shrink <code>\@colht</code> temporarily by the height-plus-depth of page-wise footnotes and the natural size of the skip above them.	192
<code>\pcol@unvbox@cclv</code> : Introduced to add stretch/shrink components of <code>\skip\footins</code> at the bottom of a column-page if the page has page-wise footnotes.	192
<code>\pcol@deferredfootins</code> : Introduced to insert deferred footnotes.	192
<code>\pcol@combinefootins</code> : Introduced to put footnotes into pre-environment stuff.	193
<code>\pcol@putfootins</code> : Introduced to put page-wise footnotes to a page.	193
<code>\pcol@sync</code> : Revise reflecting the redesign of page context.	201
<code>\pcol@sync</code> : Add pre-flushing column height check taking page-wise footnotes into account.	201
<code>\pcol@flushcolumn</code> : Revise reflecting the redesign of <code>\pcol@getcurrfoot</code>	204
<code>\pcol@flushcolumn</code> : Add operations for page-wise footnotes.	204
<code>\pcol@flushcolumn</code> : Add the examination of $\kappa_c(\rho) = \infty$ to cope with a rare-case interaction of pre-flushing column height check and float columns in last pages.	204
<code>\pcol@flushcolumn</code> : Add <code>\@colht</code> shrinking by page-wise footnotes.	206
<code>\pcol@flushcolumn</code> : Revise reflecting the redesign of page context.	206
<code>\pcol@flushcolumn</code> : Add <code>\@colht</code> shrinking by page-wise footnotes.	206
<code>\pcol@makefcolumn</code> : Encapsulate the float column in a <code>\vbox</code> of <code>\@colht</code> so that vertical skips at the top and bottom are not lost when the column is put back to the main vertical list, and makes the assignment to <code>\@currbox</code> global because the box is now referred to after <code>\output</code> is completed.	207
<code>\pcol@measurecolumn</code> : Add calculation of V_P and c_{\max}	209
<code>\pcol@synccolumn</code> : Change code structure removing the case for overflow synchronized pages.	211
<code>\pcol@synccolumn</code> : Remove <code>\penalty-10000</code> made unnecessary by the redesign of overflow synchronized pages.	211
<code>\pcol@output@flush</code> : Add <code>\pcol@Logstart</code> and <code>\pcol@Logend</code>	214
<code>\pcol@output@clear</code> : Add <code>\pcol@Logstart</code> and <code>\pcol@Logend</code>	214
<code>\pcol@makeflushedpage</code> : Revise reflecting the redesign of page context.	215
<code>\pcol@makeflushedpage</code> : Add <code>\@colht</code> shrinking by page-wise footnotes.	217
<code>\pcol@makeflushedpage</code> : Add incorporation of page-wise footnotes.	218
<code>\pcol@imakeflushedpage</code> : Revise reflecting the redesign of <code>\pcol@getcurrfoot</code>	220
<code>\pcol@imakeflushedpage</code> : Add <code>\pcol@Logstart</code> and <code>\pcol@Logend</code>	220
<code>\pcol@freshpage</code> : Add argument 0 for the invocations of <code>\pcol@startcolumn</code> to inhibit inserting deferred page-wise footnotes.	221
<code>\pcol@output@end</code> : Add incorporation of page-wise footnotes in the last ordinary pages followed by float-pages.	222
<code>\pcol@output@end</code> : Add insertion of page-wise footnotes to be merged.	225
<code>\pcol@invokeoutput</code> : Add logging.	226
<code>\pcol@zparacol</code> : Add initialization of <code>\pcol@footnotabase</code> and <code>\pcol@nfootnotes</code> , and redefinitions of <code>\footnote</code> , <code>\footnotemark</code> , <code>\footnotetext</code> and <code>\@footnotetext</code>	231
<code>\pcol@zparacol</code> : Add nullification of API macros of footnote typesetting definition.	231
<code>\pcol@ignore</code> : Introduced for nullification of API macros of footnote typesetting definition.	233
<code>\pcol@switchcol</code> : Add column-scanning and page-overflow check for synchronized column-switching.	247
<code>\pcol@visitalcols</code> : Introduced for column-scanning in synchronized column-switching and page flushing.	248
<code>\pcol@flushclear</code> : Introduced for column-scanning in synchronized column-switching and page flushing.	253
General: Add the subsection “Commands for Footnotes” to describe newly introduced macros for page-wise footnotes.	253
<code>\multicolumnfootnotes</code> : Introduced to declare the default column-wise footnote typesetting explicitly.	253

<code>\singlecolumnfootnotes</code> : Introduced to declare the page-wise but non-merged footnote typesetting.	253
<code>\mergedfootnotes</code> : Introduced to declare the page-wise and merged footnote typesetting.	253
<code>\pcol@fntext</code> : Introduced for footnote encapsulation and deferring.	254
<code>\pcol@fntexttop</code> : Introduced for footnote encapsulation.	254
<code>\pcol@fntextother</code> : Introduced for footnote encapsulation and deferring.	254
<code>\pcol@fntextbody</code> : Introduced for footnote encapsulation and height capping.	255
<code>\fncounteradjustment</code> : Introduced to make <code>footnote</code> counter is consistent with its origin at the beginning of <code>paracol</code> and the number of footnotes given in the environment at its end.	255
<code>\nofncounteradjustment</code> : Introduced to disable the footnote counter adjustment.	255
<code>\pcol@footnoterule</code> : Introduced to keep the original definition of <code>\footnoterule</code>	256
<code>\pcol@@footnote</code> : Introduced to keep the original definition of <code>\footnote</code>	256
<code>\pcol@@footnotemark</code> : Introduced to keep the original definition of <code>\footnotemark</code>	256
<code>\pcol@footnotetext</code> : Introduced to keep the original definition of <code>\footnotetext</code>	256
<code>\pcol@footnote</code> : Introduced for <code>\footnote*</code> and footnote counter adjustment.	256
<code>\pcol@iffootnote</code> : Introduced for <code>\footnote*</code> and footnote counter adjustment.	256
<code>\pcol@footnotemark</code> : Introduced for <code>\footnotemark*</code> and footnote counter adjustment.	256
<code>\pcol@iffootnotemark</code> : Introduced for <code>\footnotemark*</code> and footnote counter adjustment.	256
<code>\pcol@adjustfnctr</code> : Introduced for <code>\footnote*</code> and <code>\footnotemark*</code>	256
<code>\pcol@iadjustfnctr</code> : Introduced for <code>\footnote*</code> and <code>\footnotemark*</code>	256
<code>\pcol@calcfnctr</code> : Introduced for <code>\footnote*</code> , <code>\footnotemark*</code> and <code>\footnotetext*</code>	256
<code>\pcol@footnotetext</code> : Introduced for <code>\footnotetext*</code>	257
<code>\pcol@iffootnotetext</code> : Introduced for <code>\footnotetext*</code>	257
<code>\pcol@iifootnotetext</code> : Introduced for <code>\footnotetext*</code>	257
<code>\endparacol</code> : Add pre-flushing column height check and footnote counter adjustment.	268
v1.2-3	
General: Fix a problem in synchronization. (2013/05/11)	1
<code>\ifpcol@flush</code> : Introduced to fix the problem with a too-tall page at synchronization.	132
<code>\pcol@sync</code> : Modify the action on the page overflow to return from <code>\output</code> without flushing so that the page is broken outside <code>\output</code> to place top floats above the synchronization point set in the next page.	201
v1.2-4	
General: Add column-swapping functions. (2013/05/11)	1
General: Add description of <code>\[no]swapcolumninevenpages</code>	22
General: Add the subsection “Column-Swapping”	74
<code>\ifpcol@swapcolumn</code> : Introduced for column-swapping in even pages.	135
<code>\pcol@outputelt</code> : Add column-swapping for even pages if specified.	155
<code>\pcol@imakeflushedpage</code> : Add column-swapping for even pages if specified.	220
<code>\pcol@flushfloats</code> : Add column-swapping for even pages if specified.	220
<code>\pcol@zparacol</code> : Modify the setting of <code>\if@firstcolumn</code> according to column-swapping.	229
<code>\pcol@zparacol</code> : Add nullification of <code>\[no]swapcolumninevenpages</code>	231
<code>\pcol@ignore</code> : Introduced for nullification of <code>\[no]swapcolumninevenpages</code>	233
<code>\pcol@sptext</code> : Add <code>\pcol@swapcolumn</code> for column-swapping.	247
<code>\pcol@switchcol</code> : Add <code>\pcol@swapcolumn</code> for column-swapping to turn <code>\if@firstcolumn</code>	247
General: Add the section “Column-Swapping” to describe newly introduced macros for column-swapping.	258
<code>\swapcolumninevenpages</code> : Introduced to enable column-swapping.	258
<code>\noswapcolumninevenpages</code> : Introduced to disable column-swapping.	258
<code>\pcol@swapcolumn</code> : Introduced to convert column ordinal and its position.	259
v1.2-5	
General: Allow a <code>paracol</code> environment is enclosed in list-like environments. (2013/05/11)	1
General: Add an item to show that <code>paracol</code> can be enclosed in a <code>list</code> -like environment.	15

General: Modify the description about <code>\linewidth</code> reflecting the fact that <code>paracol</code> may be included in a <code>list</code> -like environment.	20
<code>\pcol@lrmargin</code> : Introduced to let <code>\linewidth</code> for each column has the value according to w_c and the list-like environment surrounding <code>paracol</code> environment.	137
<code>\pcol@getcurrcol</code> : Move assignment w_c to <code>\hsize</code> and <code>\linewidth</code> to <code>\pcol@invokeoutput</code>	185
<code>\pcol@invokeoutput</code> : Move the setting of <code>\linewidth</code> and <code>\hsize</code> from <code>\pcol@getcurrcol</code> to this macro and add <code>\parshape</code>	226
<code>\pcol@zparacol</code> : Remove the setting <code>\columnwidth</code> , <code>\hsize</code> and <code>\linewidth</code> because they are properly set in and after <code>\pcol@output@start</code>	229
<code>\pcol@zparacol</code> : Add the setting of <code>\pcol@lrmargin</code>	229
<code>\pcol@spstext</code> : Add setting of <code>\columnwidth</code> , <code>\linewidth</code> and <code>\parshape</code> to have indented spanning text with surrounding <code>list</code> -like environments.	247
<code>\endparacol</code> : Remove <code>\global</code> assignment of <code>\hsize</code> and <code>\linewidth</code> because assignments of them in <code>paracol</code> are now perfectly local.	268
v1.2-6	
General: Add <code>\localcounter</code> . (2013/05/11)	1
<code>\localcounter</code> : Introduced to remove the argument counter from Θ^g	242
<code>\pcol@remctrelt</code> : Recode to use newly introduced <code>\pcol@removecounter</code>	242
<code>\pcol@removecounter</code> : Introduced for the counter removal operation in <code>\localcounter</code> and <code>\pcol@remctrelt</code>	242
<code>\pcol@iremctrelt</code> : Add the first argument Θ' as the counter list from which the second argument θ is removed.	242
v1.2-7	
General: Bug fixes and minor revisions as follows. (2013/05/11)	1
General: Remove <code>\nosv</code> from verbatim example of Table 1 shown in the right column.	9
General: Correct a few words in German and English libretti.	13
General: Add the section “Known and Unknown Problems” to summarize a few typesetting issues and warn users of the possibility of bugs.	59
<code>\ifpcol@output</code> : Introduced to solve the <code>\output</code> request sneaking.	131
<code>\ifpcol@lastpagesave</code> : Introduced to fix the bug that <code>\@makecol</code> and <code>\pcol@makefcolumn</code> in <code>\pcol@flushcolumn</code> misunderstand that non-last pages are last.	132
<code>\pcol@output</code> : Add the examination of <code>\ifpcol@output</code> and L ^A T _E X’s original sequence for <code>\output</code> request sneaked from outside of <code>paracol</code> environment.	143
<code>\pcol@output</code> : Add the assignment of <code>\@maxdepth</code> to <code>\maxdepth</code> to nullify the temporary setting done by <code>\@addtobot</code>	143
<code>\pcol@makecol</code> : Introduced to cope with the careless implementation of <code>\@makecol</code> in pL ^A T _E X.	144
<code>\pcol@makecol</code> : Remove unnecessary check of <code>\ifpcol@lastpage</code> on the redefinition of <code>\@textbottom</code>	145
<code>\pcol@combinefloats</code> : Add the assignment of <code>\@maxdepth</code> to <code>\maxdepth</code> to nullify the temporary setting done by <code>\@addtobot</code>	146
<code>\pcol@@combinefloats</code> : Introduced to solve the <code>\output</code> request sneaking.	146
<code>\pcol@cflt</code> : Replace <code>\maxdepth</code> with <code>\@maxdepth</code>	148
<code>\pcol@nextpage</code> : Remove unnecessary scan of $\pi(p_t)$	150
<code>\pcol@outputelt</code> : Add <code>\boxmaxdepth = \@maxdepth</code> for depth capping.	155
<code>\pcol@output@start</code> : Add <code>\pcol@outputtrue</code> to solve the <code>\output</code> request sneaking.	174
<code>\pcol@output@start</code> : Include the effect of the separation of pre-environment bottom floats and columns in the starting page into the check of too large pre-environment stuff.	174
<code>\pcol@makenormalcol</code> : Turn <code>\ifpcol@lastpage</code> be <i>true</i> temporarily for <code>\pcol@combinefloats</code> to separate bottom floats in pre-environment stuff and the multi-column stuff in <code>paracol</code> environment by <code>\textfloatsep</code>	178
<code>\pcol@output@switch</code> : Modify broadcasting of $\kappa_c(\sigma)$ so that <code>\@afterindent</code> is broadcasted with <code>\@nobreak</code>	182

<code>\pcol@flushcolumn</code> : Save <code>\ifpcol@lastpage</code> into <code>\ifpcol@lastpagesave</code> and turn <code>\ifpcol@lastpage</code> <i>false</i> temporarily during the macro works on non-top and thus non-last pages to fix the bug that <code>\makecol</code> and <code>\pcol@makefcolumn</code> misunderstand the page they work on is last.	204
<code>\pcol@flushcolumn</code> : Replace <code>\makecol</code> with <code>\pcol@makecol</code> to cap the depth of <code>\outputbox</code> by <code>\maxdepth</code> even with pL ^A T _E X.	204
<code>\pcol@flushcolumn</code> : Add the restore of <code>\ifpcol@lastpage</code> from <code>\ifpcol@lastpagesave</code> . . .	206
<code>\pcol@makefcolumn</code> : Replace the sequence of operations to make a usual float column with <code>\toplist</code> with the newly introduced <code>\pcol@makefcolpage</code>	207
<code>\pcol@makefcolpage</code> : Introduced to implement the operations to make a float column performed in three macros.	208
<code>\pcol@synccolumn</code> : Add an shrink of 1/10000 fil to the bottom of flushed column pages to cancel finite shrinks just below synchronization points.	211
<code>\pcol@output@flush</code> : Add <code>\boxmaxdepth = \maxdepth</code> for depth capping.	214
<code>\pcol@output@flush</code> : Add <code>\boxmaxdepth = \maxdepth</code> for depth capping.	214
<code>\pcol@output@clear</code> : Add <code>\boxmaxdepth = \maxdepth</code> for depth capping.	214
<code>\pcol@imakeflushedpage</code> : Enclose the column-page building process in a group to fix the bug which lets <code>\topfigrule = \relax</code> affecting to another column.	220
<code>\pcol@imakeflushedpage</code> : Replace the sequence of operations to make a usual float column with <code>\toplist</code> with the newly introduced <code>\pcol@makefcolpage</code>	220
<code>\pcol@imakeflushedpage</code> : Replace <code>\makecol</code> with <code>\pcol@makecol</code> to cap the depth of <code>\outputbox</code> by <code>\maxdepth</code> even with pL ^A T _E X.	220
<code>\pcol@output@end</code> : Add <code>\pcol@outputfalse</code> to solve the <code>\output</code> request sneaking.	222
<code>\pcol@output@end</code> : Add <code>\boxmaxdepth = \maxdepth</code> for depth capping knowing it is redundant.	222
<code>\pcol@zparacol</code> : Add the saving of <code>\combinefloats</code>	229
<code>\globalcounter</code> : Examine if the argument counter is already in Θ^g to avoid the duplication in the list which caused a bug.	242
v1.21	
General: Fix the bug by which a column having empty column-pages followed by a synchronization point is lost or placed in a wrong page. (2013/06/06)	1
<code>\pcol@flushcolumn</code> : Add page and column numbers to logging.	204
<code>\pcol@flushcolumn</code> : Fix the bug that $\kappa_c(\beta^p)$ is let have <code>\pcol@page</code> , which can be less than $p_t = \pcol@toppage$, to cause the column c is lost or moved to a wrong page.	206
<code>\pcol@invokeoutput</code> : Add zero-clearing of <code>\deadcycles</code>	226
v1.22	
General: Fix the bug that <code>\color</code> and its relatives in a paragraph or around page top causes inconsistency of color context. (2013/06/30)	1
General: Add the subsection “Coloring in Horizontal Mode”.	71
<code>\ifpcol@output</code> : Add a user <code>\pcol@reset@color@pop</code> to inhibit uncoloring if <i>false</i>	131
<code>\ifpcol@inner</code> : Introduced to know if we are in a <code>\vbox</code>	134
<code>\pcol@everyvbox</code> : Introduced to keep <code>\everyvbox</code> work as usual while having <code>\pcol@innertrue</code> in it always.	140
<code>\pcol@output</code> : Add reset of <code>\set@color</code>	143
General: <code>\pcol@op@cpush</code> was introduced for output request to push color stack but removed in v1.34.	173
General: <code>\pcol@op@cpop</code> was introduced for output request to pop color stack but removed in v1.34.	173
General: <code>\pcol@op@cset</code> was introduced for output request to set γ_0^c but removed in v1.34 . . .	173
<code>\pcol@specialoutput</code> : Add the invocation of <code>\pcol@output@f</code> for $f \in \{cpush, cpop, cset\}$. . .	174
<code>\pcol@output@start</code> : Move emptying <code>\pcol@colorstack</code> to <code>\pcol@zparacol</code>	178
General: <code>\pcol@output@cpush</code> was introduced for color stack pushing but removed in v1.34. . .	187
General: <code>\pcol@output@icpush</code> was introduced to implement <code>\pcol@output@cpush</code> but removed in v1.34.	187

General: <code>\pcol@output@cpop</code> was introduced for color stack popping but removed in v1.34. .	187
General: <code>\pcol@reset@color@elt</code> was moved from the position where <code>\pcol@reset@color@pop</code> was defined because it became to be used only by <code>\pcol@output@cpop</code> , but removed in v1.34.	187
General: <code>\pcol@output@csset</code> was introduced to set γ_0^c but removed in v1.34.	187
General: <code>\pcol@output@icset</code> was introduced to implement <code>\pcol@output@csset</code> but removed in v1.34.	187
General: <code>\pcol@return@from@color</code> was introduced to implement <code>\pcol@output@cpush</code> , <code>\pcol@output@cpop</code> and <code>\pcol@output@csset</code> but removed in v1.34.	187
<code>\pcol@zparacol</code> : Add a trick with <code>\everyvbox</code> to turn on <code>\ifpcol@inner</code> in every <code>\vbox</code> . . .	230
<code>\pcol@zparacol</code> : Move initial emptying of Γ from <code>\pcol@output@start</code> to <code>\pcol@zparacol</code> . . .	230
<code>\pcol@zparacol</code> : Add initial emptying of $\hat{\Gamma}$ and χ	230
<code>\columncolor</code> : Add the definition of <code>\pcol@colorcommand</code> for warning in <code>\pcol@icolumncolor</code>	260
<code>\normalcolumncolor</code> : Add the definition of <code>\pcol@colorcommand</code> for warning in <code>\pcol@icolumncolor</code>	260
General: <code>\pcol@getshadowcc</code> was introduced for setting $\hat{\gamma}_0^c$ into γ_0^c locally, but removed in v1.34.	260
<code>\pcol@icolumncolor</code> : Add warning of ineffective uses of <code>\columncolor</code> and <code>\normalcolumncolor</code> , and modify the mechanism to update γ_0^c and to rewind/reestablish color stack.	260
<code>\pcol@iicolumncolor</code> : Introduced for setting γ_0^c and $\hat{\gamma}_0^c$, and pushing χ	260
<code>\pcol@set@color@push</code> : Modified to push color stack by <code>\output</code> always.	262
<code>\pcol@reset@color@pop</code> : Modified to pop color stack by <code>\output</code> and to examine if <code>\ifpcol@output = true</code>	263
General: <code>\pcol@color@invokeoutput</code> was introduced for <code>\output</code> request for coloring but removed in v1.34.	263
General: <code>\pcol@color@invokeoutput@v</code> was introduced for <code>\output</code> request for coloring but removed in v1.34.	263
<code>\pcol@restoreeveryvbox</code> : Introduced to reflect <code>\global</code> updates on <code>\everyvbox</code> in <code>paracol</code> environments.	269
v1.23	
General: Fix the problem that a colored text has a line break candidate at its end inappropriately. (2013/07/08)	1
<code>\pcol@icolumncolor</code> : Add an argument of <code>\relax</code> to <code>\pcol@color@invokeoutput</code> because any insertion after <code>\adjust</code> is not required or justified.	260
<code>\pcol@set@color@push</code> : Add an argument of <code>null \hskip</code> to <code>\pcol@color@invokeoutput</code> so that the first word of a colored text is hyphenated.	262
<code>\pcol@reset@color@pop</code> : Add an argument of <code>\relax</code> to <code>\pcol@color@invokeoutput</code> so that the last word of a colored text is not followed by a line break candidate.	263
General: <code>\pcol@color@invokeoutput</code> was modified to add second argument <code>s</code> to insert a null skip after <code>\adjust</code> only when the macro is invoked from <code>\pcol@set@color@push</code> in horizontal mode.	263
v1.24	
General: Fix the problem caused by the concealment of <code>\adjust</code> in math group. (2013/07/27) .	1
General: Add the subsection “Coloring in Math Mode”.	73
<code>\pcol@mcid</code> : Introduced for coloring specified in math mode.	130
<code>\pcol@output</code> : Add zero-clear of <code>\pcol@mcid</code>	143
General: <code>\pcol@op@mcpush</code> was introduced for coloring specified in math mode but removed in v1.34.	174
General: <code>\pcol@op@mcpush@pone</code> was introduced for coloring specified in math mode but removed in v1.34.	174
General: <code>\pcol@op@mcpop</code> was introduced for coloring specified in math mode but removed in v1.34.	174

General: <code>\pcol@op@mcpop@pone</code> was introduced for coloring specified in math mode but removed in v1.34.	174
<code>\pcol@specialoutput</code> : Add examination with P_{push} and P_{pop} and invocation of <code>\pcol@output@mcpush</code> and <code>\pcol@output@mcpop</code>	174
General: <code>\pcol@output@mcpush</code> was introduced for coloring specified in math mode but removed in v1.34.	187
General: <code>\pcol@output@imcpush</code> was introduced for coloring specified in math mode but removed in v1.34.	187
General: <code>\pcol@output@mcpop</code> was introduced for coloring specified in math mode but removed in v1.34.	187
General: <code>\pcol@output@mcpop@elt</code> was introduced for coloring specified in math mode but removed in v1.34.	187
<code>\pcol@icolumncolor</code> : Add math mode to the cases of ineffective use.	260
<code>\pcol@mcpushlimit</code> : Introduced for coloring specified in math mode.	262
<code>\pcol@set@color@push</code> : Add the mechanism special for math mode.	262
<code>\pcol@reset@color@mcpop</code> : Introduced for coloring specified in math mode.	263
v1.3-1	
General: Fix the known problem of the placement of page-crossing spanning texts. (2013/09/17)	1
General: Remove the problem description of the placement of page-crossing spanning texts because it has been solved.	59
General: Change the section title from “Column-Swapping” to “Parallel-Paging, Column-Swapping, Column-Separating Rule Drawing and Background Painting” to discuss related issues together”.	74
<code>\ifpcol@sptextstart</code> : Introduced to capture the starting point of a spanning text so that the text is split from other main vertical list stuff.	131
<code>\ifpcol@sptext</code> : Renamed from <code>\ifpcol@mctext</code> following the naming convention, and move the timing of turning <i>true</i> from the end of a spanning text to its beginning.	131
<code>\pcol@prespan</code> : Introduced to save pre-spanning-text stuff.	138
<code>\pcol@output</code> : Add <code>\ifpcol@sptextstart = false</code> to the condition for the warning of too small <code>\vsize</code>	143
<code>\pcol@makecol</code> : Add a function to capture a broken spanning text, to combine it with pre-spanning-text stuff, and to shift it left on column-swapping.	145
<code>\pcol@output@switch</code> : Add the capture of a spanning text when it is closed.	180
<code>\pcol@output@switch</code> : Rename <code>\ifpcol@mctext</code> as <code>\ifpcol@sptext</code>	182
<code>\pcol@shiftspanning</code> : Introduced to shift a spanning text to left if the column-0 is not leftmost due to column-swapping.	182
<code>\pcol@restartcolumn</code> : Rename <code>\pcol@restorecst@restart</code> as <code>\pcol@putbackmvl</code>	183
<code>\pcol@putbackmvl</code> : Renamed from <code>\pcol@restorecst@restart</code> and the operations to save pre-spanning-text stuff is added.	186
<code>\pcol@sptext</code> : Add <code>\ifpcol@sptextstart = true</code> before first synchronized column-switching to let <code>\pcol@output@switch</code> save pre-spanning-text stuff, move the timing of <code>\ifpcol@sptext = true</code> from the end of spanning text to the beginning so that <code>\output</code> routine for a page break in the text capture the pre-break portion, and remove the invocation of <code>\pcol@swapcolumn</code> because spanning texts are now always put into the column-0.	247
v1.3-2	
General: Introduce parallel-paging. (2013/09/17)	1
General: Add description of parallel-paging.	16
General: Add description of the optional argument of <code>\columnratio</code> for parallel-paging.	19
General: Add description of <code>\setcolumnwidth</code>	20
General: Add the section “Two-Sided Typesetting and Parallel-Paging”.	38
General: Add comments about the limitation of parallel-paging.	60
General: Add overview description of parallel-paging.	74

<code>\pcol@ncolleft</code> : Introduced to specify the number of columns in left parallel-pages.	129
<code>\ifpcol@output</code> : Add a user <code>\@outputpage</code> for parallel-paging.	131
<code>\ifpcol@paired</code> : Introduced for parallel-paging which has paired and non-paired mode.	135
<code>\pcol@rightpage</code> : Introduced to have the ship-out image of a right parallel-page.	138
<code>\pcol@opcol</code> : Rename <code>\pcol@outputpage</code> as <code>\pcol@outputcolumns</code>	148
<code>\pcol@setpnoelt</code> : Add an operation to increment <i>page</i> (<i>p</i>) by two for non-paired parallel-paging.	149
<code>\pcol@startpage</code> : Duplicate <code>\stepcounter</code> of <code>\c@page</code> if non-paired parallel-paging is in effect.	152
<code>\pcol@outputcolumns</code> : Rename <code>\pcol@outputpage</code> as <code>\pcol@outputcolumns</code>	154
<code>\pcol@outputelt</code> : Add building a empty right parallel float page.	155
<code>\pcol@outputelt</code> : Move the core of ship-out image building to <code>\pcol@ioutputelt</code> for parallel paging.	155
<code>\pcol@ioutputelt</code> : Introduced for parallel-paging.	156
<code>\pcol@phantom</code> : Introduced for parallel-paging.	158
<code>\pcol@outputpage</code> : Introduced keep the original definition of <code>\@outputpage</code>	160
<code>\@outputpage</code> : Redefined for parallel-paging.	160
<code>\pcol@outputpage@l</code> : Introduced for shipping out left parallel-pages.	162
<code>\pcol@outputpage@r</code> : Introduced for shipping out right parallel-pages.	162
<code>\pcol@putfootins</code> : Change users <code>\pcol@outputelt</code> to <code>\pcol@ioutputelt</code>	193
<code>\pcol@output@flush</code> : Add depth capping of <code>\pcol@rightpage</code>	214
<code>\pcol@output@clear</code> : Add depth capping of <code>\pcol@rightpage</code> and building an empty right parallel-page for each page-wise float page.	214
<code>\pcol@makeflushedpage</code> : Completely redesigned with new macro <code>\pcol@imakeflushedpage</code>	215
<code>\pcol@imakeflushedpage</code> : Introduced for parallel-paging.	218
<code>\pcol@flushfloats</code> : Completely redesigned with new macro <code>\pcol@iflushfloats</code>	220
<code>\pcol@iflushfloats</code> : Introduced for parallel-paging.	220
<code>\pcol@output@end</code> : Add parallel-paging operations.	222
<code>\paracol</code> : Modify to add the optional argument C_L and optional ‘*’ for parallel-paging.	227
<code>\pcol@xparacol</code> : Introduced to let $C_L = C$ if the optional argument C_L is not given to <code>\paracol</code>	227
<code>\pcol@yparacol</code> : Introduced to process the optional ‘*’ given with the optional argument C_L of <code>\paracol</code>	227
<code>\pcol@zparacol</code> : Introduced to add the optional argument C_L and optional ‘*’ to <code>\paracol</code> for parallel-paging and to do what had done by <code>\paracol</code>	227
<code>\pcol@zparacol</code> : Add operations to define the width of columns and column-separating gaps in right parallel-pages.	229
<code>\columnratio</code> : Add optional second argument for fractions in right parallel-pages.	234
<code>\pcol@icolumnratio</code> : Introduced to process the optional second argument of <code>\columnratio</code>	234
<code>\pcol@columnratioleft</code> : Renamed from <code>\pcol@columnratio</code> to clarify it has fractions for left parallel-pages.	234
<code>\pcol@columnratioreight</code> : Introduced to keep column width fractions for right parallel-pages.	234
<code>\pcol@setcolwidth@r</code> : Add arguments C^0 , C^1 , (<i>ratio</i>) and (<i>spec</i>) for columns in right parallel-pages.	235
<code>\pcol@swapcolumn</code> : Add two arguments C^0 and C^1 as the third and fourth ones to modify the calculation of c_2 with them for column-swapping with parallel-paging.	259
v1.3-3	
General: Introduce column-separating rule drawing and background painting. (2013/09/17)	1
General: Add the sub-section “Commands for Two-Sided Typesetting and Marginal Note Placement”.	21
General: Add description of <code>\twosided</code>	21
General: Rename the sub-section title from “Commands for Text Coloring” to “Commands for Coloring Texts and Column-Separating Rules” to add description of the rule coloring together with the rule drawing itself.	25

General: Add description of <code>\coloredwordhyphenated</code> and <code>\nocoloredwordhyphenated</code>	26
General: Add description of <code>\colseprulecolor</code> and <code>\normalcolseprulecolor</code>	26
General: Add the sub-section “Commands for Background Painting.	27
General: Add description of <code>\backgroundcolor</code>	27
General: Add description of <code>\nbackgroundcolor</code> and <code>\resetbackgroundcolor</code>	28
General: Add description of <code>\pagerim</code>	29
General: Remove the problem description of the lack of column-separating rule drawing because it has been implemented.	59
General: Add comments about the imperfectness of extension of background painting regions.	60
General: Add $\pi^s(p)$ to the page context of p for column-separating rule drawing and background painting.	66
General: Add overview description of column-separating rule drawing and background painting.	74
<code>\pcol@ncol</code> : Add initial zero-clearing for safe reference in <code>\@outputpage</code> invoked prior to the first <code>paracol</code>	129
<code>\ifpcol@output</code> : Add a user <code>\@outputpage</code> for background-painting.	131
<code>\ifpcol@firstpage</code> : Introduced to know if a spanning stuff is pre-environment one.	134
<code>\ifpcol@havelastpage</code> : Introduced to know if a page to be put has the last page of a <code>paracol</code> environment.	134
<code>\ifpcol@bg@swap</code> : Introduced for mirrored background painting for even numbered pages.	135
<code>\ifpcol@bg@@swap</code> : Introduced for mirrored background painting for even numbered pages.	135
<code>\ifpcol@bg@painted</code> : Introduced to examine if a set of regions are painted.	136
<code>\pcol@bg@leftmargin</code> : Introduced for background painting.	137
<code>\pagerim</code> : Introduced to specify the page rim size for background coloring.	138
<code>\pcol@tempboxa</code> : Introduced to have materials temporarily for column-separating rule drawing or background painting.	139
<code>\pcol@makecol</code> : Add the addition of the element to $\pi^s(p)$ for a broken spanning text.	145
<code>\pcol@makecol</code> : Add <code>\colht</code> and <code>\relax</code> as the first and third argument of <code>\pcol@shrinkcolbyfn</code>	145
<code>\pcol@setpnoelt</code> : Revise reflecting the new page context element $\pi^s(p)$, and add a invoker <code>\pcol@makecol</code>	149
<code>\pcol@defcurrpage</code> : Revise reflecting the new page context element $\pi^s(p)$, and add a invoker <code>\pcol@makecol</code>	150
<code>\pcol@nextpelt</code> : Revise reflecting the new page context element $\pi^s(p)$, and add a invoker <code>\pcol@makecol</code>	150
<code>\pcol@getcurrpinfo</code> : Revise reflecting the new page context element $\pi^s(p)$, and add a invoker <code>\pcol@makecol</code>	150
<code>\pcol@startpage</code> : Revise reflecting the new page context element $\pi^s(p)$	152
<code>\pcol@outputelt</code> : Revise reflecting the new page context element $\pi^s(p)$	154
<code>\pcol@outputelt</code> : Add painting of page-wise float page.	155
<code>\pcol@ioutputelt</code> : Add column-separating rule drawing and background painting.	156
<code>\pcol@buildcolseprule</code> : Introduced for column-separating rule drawing and background painting for columns, column-separating gaps and spanning texts.	158
<code>\pcol@buildcselet@S</code> : Introduced for background under-painting for spanning texts.	158
<code>\pcol@buildcselet</code> : Introduced for column-separating rule drawing and background painting for columns, column-separating gaps and spanning texts.	158
<code>\pcol@hfil</code> : Introduced for column-separating rule drawing.	160
<code>\pcol@@outputpage</code> : Introduced keep the original definition of <code>\@outputpage</code>	160
<code>\@outputpage</code> : Redefined for background painting.	160
<code>\pcol@outputpage@l</code> : Introduced for column-separating rule drawing and background painting in left parallel-pages.	162
<code>\pcol@outputpage@r</code> : Introduced for column-separating rule drawing and background painting in right parallel-pages.	162
<code>\pcol@outputpage@ev</code> : Introduced for background painting.	162

\pcol@startcolumn: Add \@colht and \@tempdimb as the first and third argument of \pcol@shrinkcolbyfn.	164
\pcol@bg@from: Introduced for background painting.	165
\pcol@bg@to: Introduced for background painting.	165
\pcol@bg@paintpage: Introduced for background painting.	166
\pcol@bg@@paintpage: Introduced for background painting.	166
\pcol@bg@paintcolumns: Introduced for background painting.	166
\pcol@bg@@paintcolumns: Introduced for background painting.	166
\pcol@bg@paintbox: Introduced for background painting.	166
\pcol@bg@@paintbox: Introduced for background painting.	166
\pcol@bg@paint@i: Introduced for background painting.	166
\pcol@bg@paint@ii: Introduced for background painting.	167
\pcol@bg@swappage: Introduced for background painting.	167
\pcol@bg@paintregion@i: Introduced for background painting.	168
\pcol@bg@calculate: Introduced for background painting.	169
\pcol@bg@advance: Introduced for background painting.	169
\pcol@bg@negative: Introduced for background painting.	169
\pcol@bg@nadvance: Introduced for background painting.	169
\pcol@bg@dimen: Introduced for background painting.	169
\pcol@bg@addext: Introduced for background painting.	169
\pcol@bg@ext@inf@l: Introduced for background painting.	169
\pcol@bg@ext@inf@r: Introduced for background painting.	169
\pcol@bg@ext@inf@t: Introduced for background painting.	169
\pcol@bg@ext@inf@b: Introduced for background painting.	169
\pcol@bg@paperwidth: Introduced for background painting.	170
\pcol@bg@paperheight: Introduced for background painting.	170
\pcol@bg@pageleft: Introduced for background painting.	170
\pcol@bg@pagetop: Introduced for background painting.	170
\pcol@bg@textheight: Introduced for background painting.	170
\pcol@bg@columnleft: Introduced for background painting.	170
\pcol@bg@columnright: Introduced for background painting.	170
\pcol@bg@columnwidth: Introduced for background painting.	170
\pcol@bg@columnsep: Introduced for background painting.	170
\pcol@bg@preposttop: Introduced for background painting.	171
\pcol@bg@preposttop@left: Introduced for background painting.	171
\pcol@bg@preposttop@right: Introduced for background painting.	171
\pcol@bg@columnntop: Introduced for background painting.	171
\pcol@bg@columnheight: Introduced for background painting.	171
\pcol@bg@floatheight: Introduced for background painting.	171
\pcol@bg@footnoteheight: Introduced for background painting.	171
\pcol@bg@spanningtop: Introduced for background painting.	171
\pcol@bg@spanningheight: Introduced for background painting.	171
\pcol@bg@@c: Introduced for background painting.	172
\pcol@bg@@C: Introduced for background painting.	172
\pcol@bg@@g: Introduced for background painting.	172
\pcol@bg@@G: Introduced for background painting.	172
\pcol@bg@@s: Introduced for background painting.	172
\pcol@bg@@S: Introduced for background painting.	172
\pcol@bg@@t: Introduced for background painting.	172
\pcol@bg@@T: Introduced for background painting.	172
\pcol@bg@@b: Introduced for background painting.	172
\pcol@bg@@B: Introduced for background painting.	172
\pcol@bg@@l: Introduced for background painting.	172

<code>\pcol@bg@L</code> : Introduced for background painting.	172
<code>\pcol@bg@R</code> : Introduced for background painting.	172
<code>\pcol@bg@R</code> : Introduced for background painting.	172
<code>\pcol@bg@f</code> : Introduced for background painting.	172
<code>\pcol@bg@F</code> : Introduced for background painting.	172
<code>\pcol@bg@n</code> : Introduced for background painting.	172
<code>\pcol@bg@N</code> : Introduced for background painting.	172
<code>\pcol@bg@p</code> : Introduced for background painting.	172
<code>\pcol@bg@P</code> : Introduced for background painting.	172
<code>\pcol@output@start</code> : Let <code>\ifpcol@output = false</code> temporarily before the invocation of <code>\@outputpage</code> for too tall pre-environment stuff because the page is considered as outside <code>paracol</code> environments.	174
<code>\pcol@output@start</code> : Add background painting of pre-environment stuff.	176
<code>\pcol@output@switch</code> : Add $\pi^s(p) = \text{\pcol@sptextlist}$ to the argument of <code>\pcol@defcurrpage</code>	180
<code>\pcol@restartcolumn</code> : Add <code>\@colht</code> and <code>\@tempdimb</code> as the first and third argument of <code>\pcol@shrinkcolbyfn</code>	183
<code>\pcol@shrinkcolbyfn</code> : Add the first argument <i>height</i> for the user <code>\pcol@ioutputelt</code>	192
<code>\pcol@flushcolumn</code> : Add <code>\@colht</code> and <code>\relax</code> as the first and third argument of <code>\pcol@shrinkcolbyfn</code> for all of three invocations of it.	204
<code>\pcol@hdflelt</code> : Add a user <code>\pcol@makecol</code>	210
<code>\pcol@output@clear</code> : Add background painting of float pages.	214
<code>\pcol@makeflushedpage</code> : Add background painting of page-wise floats, and a part of operations for column-separation rule drawing and background painting of page-wise footnotes.	215
<code>\pcol@makeflushedpage</code> : Add <code>\@colht</code> and <code>\relax</code> as the first and third argument of <code>\pcol@shrinkcolbyfn</code> for all of three invocations of it.	217
<code>\pcol@imakeflushedpage</code> : Implement column-separating rule and background painting of columns, column-separating gaps, spanning texts and page-wise footnotes.	218
<code>\pcol@iflushfloats</code> : Implement column-separating rule and background painting of columns and column-separating gaps	220
<code>\pcol@output@end</code> : Add background painting of page-wise footnotes and setting of <code>\pcol@preposttop</code>	222
<code>\pcol@output@end</code> : Add settings for background painting of post-environment stuff.	224
<code>\pcol@output@end</code> : Add background painting of page-wise footnotes.	225
<code>\pcol@zparacol</code> : Add definition of painting macros dependent to the availability of a coloring package.	230
<code>\pcol@zparacol</code> : Add new API inactivation for <code>\pcol@twosided</code>	231
<code>\pcol@setcolwidth@r</code> : <code>\columnsep</code>	235
<code>\twosided</code> : Add two-sided background painting.	258
<code>\pcol@twosided@b</code> : Introduced for two-sided background painting.	258
<code>\pcol@swapcolumn</code> : Add the assignment of <code>\pcol@colsepid</code> to let it have $c_2 - 1$ if swapped or c_2 otherwise.	259
<code>\pcol@colsepid</code> : Introduced to be let have $c_2 - 1$ if swapped or c_2 otherwise by <code>\pcol@swapcolumn</code>	259
General: Add the subsection “Commands for Column-Separating Rule Color and Background Painting” to describe newly introduced API macros to specify colors of column-separating rules and background painting.	264
<code>\colseprulecolor</code> : Introduced to specify the colors of column-separating rules.	264
<code>\pcol@defcseprulecolor@x</code> : Introduced to implement <code>\colseprulecolor</code>	264
<code>\pcol@defcseprulecolor@y</code> : Introduced to implement <code>\colseprulecolor</code>	264
<code>\pcol@defcseprulecolor</code> : Introduced to implement <code>\colseprulecolor</code>	264
<code>\normalcolseprulecolor</code> : Introduced to specify that color of column-separating rules is normal.	264

<code>\pcol@defcseprulecolor@i</code> : Introduced to implement <code>\colseprulecolor</code> and <code>\normalcolseprulecolor</code>	264
<code>\pcol@colseprulecolor</code> : Introduced to keep the color for all column-separating rules.	264
<code>\backgroundcolor</code> : Introduced to define colors for background painting.	265
<code>\nbackgroundcolor</code> : Introduced to undefine colors for background painting.	265
<code>\pcol@backgroundcolor@e</code> : Introduced to implement <code>\backgroundcolor</code> and <code>\nbackgroundcolor</code>	265
<code>\pcol@backgroundcolor</code> : Introduced to implement <code>\backgroundcolor</code> and <code>\nbackgroundcolor</code>	265
<code>\pcol@backgroundcolor@i</code> : Introduced to implement <code>\backgroundcolor</code> and <code>\nbackgroundcolor</code>	265
<code>\pcol@bg@region</code> : Introduced to implement <code>\backgroundcolor</code> and <code>\nbackgroundcolor</code>	265
<code>\pcol@backgroundcolor@ii</code> : Introduced to implement <code>\backgroundcolor</code> and <code>\nbackgroundcolor</code>	265
<code>\pcol@backgroundcolor@iii</code> : Introduced to implement <code>\backgroundcolor</code>	265
<code>\pcol@backgroundcolor@iv</code> : Introduced to implement <code>\backgroundcolor</code>	265
<code>\pcol@backgroundcolor@v</code> : Introduced to implement <code>\backgroundcolor</code>	265
<code>\pcol@backgroundcolor@x</code> : Introduced to implement <code>\backgroundcolor</code>	266
<code>\pcol@backgroundcolor@y</code> : Introduced to implement <code>\backgroundcolor</code>	266
<code>\pcol@backgroundcolor@z</code> : Introduced to implement <code>\nbackgroundcolor</code>	267
<code>\pcol@backgroundcolor@w</code> : Introduced to implement <code>\backgroundcolor</code>	267
<code>\pcol@backgroundcolor@wi</code> : Introduced to implement <code>\backgroundcolor</code> and <code>\nbackgroundcolor</code>	267
<code>\pcol@bg@color@xx</code> : Introduced to implement <code>\backgroundcolor</code> and <code>\nbackgroundcolor</code>	267
<code>\pcol@bg@mayhavecol@c</code> : Introduced to implement <code>\backgroundcolor</code> and <code>\nbackgroundcolor</code>	267
<code>\pcol@bg@mayhavecol@C</code> : Introduced to implement <code>\backgroundcolor</code> and <code>\nbackgroundcolor</code>	267
<code>\pcol@bg@mayhavecol@g</code> : Introduced to implement <code>\backgroundcolor</code> and <code>\nbackgroundcolor</code>	267
<code>\pcol@bg@mayhavecol@G</code> : Introduced to implement <code>\backgroundcolor</code> and <code>\nbackgroundcolor</code>	267
<code>\pcol@bg@defext</code> : Introduced to implement <code>\backgroundcolor</code>	267
<code>\resetbackgroundcolor</code> : Introduced to disable background-painting for all regions.	268
<code>\pcol@resetbackgroundcolor</code> : Introduced to implement <code>\resetbackgroundcolor</code>	268
<code>\pcol@bg@defined</code> : Introduced to implement <code>\resetbackgroundcolor</code>	268
v1.3-4	
General: Introduce API for column/gap width and marginal note position specification. (2013/09/17)	1
General: Add the sub-section “Commands for Two-Sided Typesetting and Marginal Note Placement”.	21
General: Add description of <code>\twosided</code>	21
General: Add description of <code>\marginparthreshold</code>	22
General: Add the section “Two-Sided Typesetting and Parallel-Paging”.	38
General: Remove μ for <code>\@mparbottom</code> from column-context because it is now in page context.	64
General: Add $\pi^m(p)$ to the page context of p for marginal note placement.	66
<code>\ifpcol@swapmarginpar</code> : Introduced for marginal-note-swapping in even pages.	135
<code>\pcol@makecol</code> : Add $\pi^m(p) = \text{\pcol@mparbottom}$ to the argument of <code>\pcol@defcurrpage</code>	145
<code>\pcol@setpnoelt</code> : Revise reflecting the new page context element $\pi^m(p)$	149
<code>\pcol@defcurrpage</code> : Revise reflecting the new page context element $\pi^m(p)$	150
<code>\pcol@nextpelt</code> : Revise reflecting the new page context element $\pi^m(p)$	150
<code>\pcol@getcurrpage</code> : Add a user <code>\pcol@addmarginpar</code>	150
<code>\pcol@getcurrpinfo</code> : Revise reflecting the new page context element $\pi^m(p)$	150

<code>\pcol@floatplacement</code> : Remove clearing operation on <code>\@mparbottom</code> because it is no longer in column-context.	151
<code>\pcol@startpage</code> : Revise reflecting the new page context element $\pi^m(p)$	152
<code>\pcol@outputelt</code> : Revise reflecting the new page context element $\pi^m(p)$	154
<code>\pcol@ioutputelt</code> : Add a logic to cope with non-uniform column-separating gaps.	156
<code>\pcol@buildcolseprule</code> : Introduced for non-uniform column-separating gaps.	158
<code>\pcol@buildcself</code> : Introduced for non-uniform column-separating gaps.	158
<code>\pcol@hfil</code> : Introduced for non-uniform column-separating gaps.	160
<code>\pcol@@outputpage</code> : Introduced keep the original definition of <code>\@outputpage</code>	160
<code>\@outputpage</code> : Redefined for marginal note placement.	160
<code>\pcol@output@start</code> : Add initialization of $\pi^m(0)$	176
<code>\pcol@output@switch</code> : Add $\pi^m(p) = \text{\pcol@mparbottom}$ to the argument of <code>\pcol@defcurrpage</code>	180
<code>\pcol@iigetcurrcol</code> : Remove the argument for $\kappa_c(\mu) = \text{\@mparbottom}$ because it is no longer in the column context.	185
<code>\pcol@setcurrcol</code> : Remove $\kappa_c(\mu) = \text{\@mparbottom}$ from the body of <code>\pcol@col:c</code> because it is no longer in the column context.	186
General: Add this section “Marginal Notes”.	194
<code>\@addmarginpar</code> : Made <code>\let</code> -equal to <code>\pcol@addmarginpar</code> in <code>paracol</code> environments.	194
<code>\pcol@addmarginpar</code> : Introduced to make a margin sharable by marginal notes from different columns.	194
<code>\pcol@@addmarginpar</code> : Introduced to keep the original definition of <code>\@addmarginpar</code>	194
<code>\pcol@getmparbottom</code> : Introduced to find the space where a marginal note is placed.	197
<code>\pcol@getmparbottom@i</code> : Introduced to find the space where a marginal note is placed.	197
<code>\pcol@getmpbelt</code> : Introduced to find the space where a marginal note is placed.	197
<code>\pcol@setmpbelt</code> : Introduced to update $\pi^m(p)$	198
<code>\pcol@setmpbelt@i</code> : Introduced to update $\pi^m(p)$	198
<code>\pcol@mparbottom@zero</code> : Introduced to give the default of <code>\pcol@mparbottom@out</code>	198
<code>\pcol@mparbottom@out</code> : Introduced to keep the last elements of $\pi^m(p_t)$ at <code>\end{paracol}</code>	198
<code>\pcol@do@mpbout</code> : Introduced to do specified operations on \mathcal{M} and its element M_L^x according to the side margin for marginal notes outside <code>paracol</code> environments.	199
<code>\pcol@do@mpbout@i</code> : Introduced to do specified operations on \mathcal{M} and its element M_L^x according to the side margin for marginal notes outside <code>paracol</code> environments.	199
<code>\pcol@do@mpbout@whole</code> : Introduced to do a specified operation on \mathcal{M}	199
<code>\pcol@do@mpbout@elem</code> : Introduced to do a specified operation on an element M_L^x in \mathcal{M}	199
<code>\pcol@bias@mpbout</code> : Introduced to perform coordinate transformation of the elements in \mathcal{M}	199
<code>\pcol@bias@mpbout@i</code> : Introduced to perform coordinate transformation of the elements in \mathcal{M}	199
<code>\pcol@getmparbottom@last</code> : Introduced to let \mathcal{M} have the occupancy information of the bottom marginal note in each margin.	200
<code>\pcol@getmparbottom@last@i</code> : Introduced to let \mathcal{M} have the occupancy information of the bottom marginal note in each margin.	200
<code>\pcol@do@mpb@all</code> : Introduced to implement <code>\pcol@bias@mpbout</code> and <code>\pcol@getmparbottom@last</code>	200
<code>\pcol@do@mpb@all@i</code> : Introduced to implement <code>\pcol@bias@mpbout</code> and <code>\pcol@getmparbottom@last</code>	200
<code>\pcol@do@mpb@all@ii</code> : Introduced to implement <code>\pcol@bias@mpbout</code> and <code>\pcol@getmparbottom@last</code>	200
<code>\pcol@imakeflushedpage</code> : Implement variable-width column-separating gaps.	218
<code>\pcol@iflushfloats</code> : Implement variable-width column-separating gaps.	220
<code>\pcol@output@end</code> : Add operations to pass \mathcal{M} to the next <code>paracol</code> environment and <code>\@mparbottom</code> to post environment typesetting.	222
<code>\pcol@output@end</code> : Add letting <code>\@mparbottom = 0</code> and $\mathcal{M} = \mathcal{M}_0$ for the simple empty page case.	224

<code>\pcol@zparacol</code> : Revise the mechanism to define the width of columns and column-separating gaps, and add local overriding definition of <code>\@addmarginpar</code> .	229
<code>\pcol@zparacol</code> : Add new API inactivation for <code>\pcol@twosided</code> .	231
General: Add the section “Column Width Setting” mainly to discuss the new API	
<code>\setcolumnwidth</code> .	234
<code>\setcolumnwidth</code> : Introduced to specify column widths and column-separating gaps more detailedly.	234
<code>\pcol@isetcolumnwidth</code> : Introduced to process the optional second argument of <code>\setcolumnwidth</code> .	234
<code>\pcol@colwidthspecleft</code> : Introduced to keep column width specifications for left parallel-pages.	234
<code>\pcol@colwidthspecright</code> : Introduced to keep column width specifications for right parallel-pages.	234
<code>\pcol@setcolumnwidth</code> : Move original functions to <code>\pcol@setcolumnwidth@r</code> and redefine this macro to switch <code>\pcol@setcolumnwidth@r</code> and <code>\pcol@setcolumnwidth@s</code> .	235
<code>\pcol@setcolwidth@r</code> : Renamed from <code>\pcol@setcolumnwidth</code> to make it clear what the macro works for.	235
<code>\pcol@setcolwidth@s</code> : Introduced to calculate w_c and g_c .	236
<code>\pcol@setcw@c</code> : Introduced to process column width components in the argument $\langle spec \rangle$ of <code>\pcol@setcolwidth@s</code> .	236
<code>\pcol@setcw@s</code> : Introduced to process column-separating gap components in the argument $\langle spec \rangle$ of <code>\pcol@setcolwidth@s</code> .	236
<code>\pcol@setcw@filunit</code> : Introduced to define the unit of infinite stretch factors in the argument $\langle spec \rangle$ of <code>\pcol@setcolwidth@s</code> .	236
<code>\pcol@setcw@scan</code> : Introduced to scan the argument $\langle spec \rangle$ of <code>\pcol@setcolwidth@s</code> .	237
<code>\pcol@setcw@getspec</code> : Introduced to parse an element in the argument $\langle spec \rangle$ of <code>\pcol@setcolwidth@s</code> .	237
<code>\pcol@setcw@getspec@i</code> : Introduced to parse an element in the argument $\langle spec \rangle$ of <code>\pcol@setcolwidth@s</code> .	237
<code>\pcol@setcw@fill</code> : Introduced to extract <code>\fill</code> factor from an element in the argument $\langle spec \rangle$ of <code>\pcol@setcolwidth@s</code> .	237
<code>\pcol@setcw@accumwd</code> : Introduced to accumulate natural and fill factors of the element in the argument $\langle spec \rangle$ of <code>\pcol@setcolwidth@s</code> .	238
<code>\pcol@setcw@set</code> : Introduced to define w_c and g_c according to the element in the argument $\langle spec \rangle$ of <code>\pcol@setcolwidth@s</code> .	238
<code>\pcol@setcw@calcfactors</code> : Introduced to calculate scaling and stretch factors from the argument $\langle spec \rangle$ of <code>\pcol@setcolwidth@s</code> .	239
<code>\pcol@setcw@calcf</code> : Introduced to calculate scaling or stretch factor from the argument $\langle spec \rangle$ of <code>\pcol@setcolwidth@s</code> .	239
<code>\pcol@setcw@scale</code> : Introduced to have the scaling factor of w_c and g_c .	239
<code>\pcol@defkw</code> : Introduced to define <code>\pcol@kw@k</code> where $k \in \{\text{pt, plus, minus, fil}\}$.	240
<code>\pcol@kw@pt</code> : Introduced to have the keyword <code>pt</code> .	240
<code>\pcol@kw@plus</code> : Introduced to have the keyword <code>plus</code> .	240
<code>\pcol@kw@minus</code> : Introduced to have the keyword <code>minus</code> .	240
<code>\pcol@kw@fil</code> : Introduced to have the keyword <code>fil</code> .	240
<code>\pcol@def@extract@fil</code> : Introduced to define <code>\pcol@extract@fil</code> .	241
<code>\pcol@extract@fil</code> : Introduced to extract infinite stretch factor from a skip if any.	241
<code>\pcol@extract@fil@i</code> : Introduced to extract the factor of <code>fil</code> from the stretch factor in a skip if any.	241
<code>\pcol@extract@fil@ii</code> : Introduced to extract the factor of <code>fil</code> from the stretch factor in a skip if any.	241
<code>\pcol@def@extract@fil@iii</code> : Introduced to define <code>\pcol@extract@fil@iii</code> .	241
<code>\pcol@extract@fil@iii</code> : Introduced to extract the factor of <code>fil</code> from the stretch factor in a skip.	241

<code>\pcol@def@extract@pt</code> : Introduced to define <code>\pcol@extract@pt</code>	241
<code>\pcol@extract@pt</code> : Introduced to extract the factor of <code>pt</code> from the <code>\the</code> representation of a dimension.	241
<code>\pcol@switchcol</code> : Remove setting <code>\if@firstcolumn</code> and the invocation of <code>\pcol@swapcolumn</code> for it because the position of marginal notes are now controlled by <code>\pcol@addmarginpar</code>	247
<code>\twosided</code> : Add two-sided marginal note placement.	258
<code>\pcol@twosided@m</code> : Introduced for two-sided marginal note placement.	258
<code>\pcol@swapcolumn</code> : Add a user <code>\pcol@addmarginpar</code> and remove <code>\paracol</code> , <code>\pcol@sptext</code> and <code>\pcol@switchcol</code>	259
<code>\marginparthreshold</code> : Introduced to specify the smallest ordinal of columns whose marginal notes go to the right margin if not swapped.	259
<code>\pcol@marginparthreshold</code> : Introduced to implement <code>\marginparthreshold</code>	259
<code>\pcol@mpthreshold@l</code> : Introduced to keep the value specified by <code>\marginparthreshold</code> for columns in left parallel-pages.	259
<code>\pcol@mpthreshold@r</code> : Introduced to keep the value specified by <code>\marginparthreshold</code> for columns in right parallel-pages.	259
v1.3-5	
General: Introduce <code>\thecolumn</code> and <code>\ensurevspace</code> , <code>\footnotelayout</code> , <code>\twosided</code> and <code>\cleardoublepage</code> . (2013/09/17)	1
General: Add description of <code>\thecolumn</code>	18
General: Add description of <code>\ensurevspace</code>	19
General: Add the sub-section “Commands for Two-Sided Typesetting and Marginal Note Placement”.	21
General: Add description of <code>\twosided</code>	21
General: Remove description of <code>\[no]swapcolumninevenpages</code> but mention they are still available.	22
General: Rename the sub-section title from “Single-Columned Footnotes” to “Page-Wise Footnotes” following new naming.	24
General: Remove description of <code>\multicolumnfootnotes</code> , <code>\singlecolumnfootnotes</code> , <code>\mergedfootnotes</code> but mention they are still available.	25
General: Add description of <code>\cleardoublepage</code>	29
General: Rename the section title from “Numbering and Placement of Single-Columned Footnotes” to “Numbering and Placement of “Page-Wise Footnotes” following new naming.	30
General: Add the section “Two-Sided Typesetting and Parallel-Paging”.	38
General: Change the subsection title from “Coloring” to “Text Coloring” to distinguish it from background painting clearly.	70
<code>\pcol@zparacol</code> : Add new API inactivation for <code>\footnotelayout</code>	231
<code>\thecolumn</code> : Introduced to let users know which column they are working in.	233
<code>\pcol@localcommands</code> : Add <code>\@elt{cleardoublepage}</code> for <code>\cleardoublepage</code>	233
<code>\pcol@switchcol</code> : Add setting $V_E = \pcol@ensurevspace$ and reinitialization of <code>\pcol@ensurevspace</code> for avoidance of post-synchronization inconsistent page break.	247
<code>\ensurevspace</code> : Introduced to declare the minimum space V_E below a synchronization point to let it stay in a page.	250
<code>\pcol@ensurevspace</code> : Introduced to keep V_E declared by <code>\ensurevspace</code>	250
<code>\pcol@@ensurevspace</code> : Introduced to pass V_E declared by <code>\ensurevspace</code> to <code>\output</code> routine.	250
<code>\cleardoublepage</code> : Added as a member of local commands and made <code>\let</code> -equal to <code>\pcol@com@cleardoublepage</code>	252
<code>\pcol@com@cleardoublepage</code> : Introduced as the implementation of <code>\cleardoublepage</code>	252
<code>\footnotelayout</code> : Introduced for easier declaration of footnote layout.	253
<code>\pcol@fnlayout@c</code> : Introduced for easier declaration of column-wise footnotes.	253
<code>\pcol@fnlayout@p</code> : Introduced for easier declaration of page-wise footnotes.	253
<code>\pcol@fnlayout@m</code> : Introduced for easier declaration of merged footnotes.	253
General: Rename the section title from “Column-Swapping” to “Two-Sided Typesetting”.	258

<code>\twosided</code> : Introduced as an easier API for various two-sided typesetting.	258
<code>\pcol@twosided</code> : Introduced to implement <code>\twosided</code>	258
<code>\pcol@twosided@p</code> : Introduced to implement <code>\twosided</code> with <code>[p]</code>	258
<code>\pcol@twosided@c</code> : Introduced to implement <code>\twosided</code> with <code>[c]</code>	258
<code>\pcol@twosided@m</code> : Introduced to implement <code>\twosided</code> with <code>[m]</code>	258
<code>\pcol@twosided@b</code> : Introduced to implement <code>\twosided</code> with <code>[b]</code>	258
v1.3-6	
General: Fix a few problems mainly related to synchronization and ordinary footnotes. (2013/09/17)	1
General: Add comments about usage of <code>\paragraph</code> etc. in spanning texts.	59
General: Change the title from “Single-Columned and Merged Footnotes” to “Page-Wise and Merged Footnotes” according to the new naming.	68
<code>\ifpcol@bfbottom</code> : Introduced to know which column-wise footnotes or bottom floats are put at the bottom of a column.	136
<code>\ifpcol@dfloats</code> : Introduced to know if the last page has deferred column-wise floats.	136
<code>\pcol@ShowBox</code> : Change <code>\unvcopy</code> to <code>\copy</code> to make sure the argument box causes overfull if its height is positive and even if it has nothing.	140
<code>\pcol@makecol</code> : Add an argument d to be assigned to <code>\boxmaxdepth</code> to let it have 0 rather than <code>\@maxdepth</code> for last page.	144
<code>\pcol@combinefloats</code> : Add special operations for columns having synchronization point to move the infinite stretch and shrink to let it follow bottom floats rather than preceding them.	146
<code>\pcol@nextpelt</code> : Fix the bug that $\pi^h(q)$ is not referred correctly.	150
<code>\pcol@output@start</code> : Change the page builder for too tall pre-environment stuff from <code>\pcol@makenormalcol</code> to <code>\@makecol</code> because the page should be built by the ordinary mechanism.	174
<code>\pcol@output@start</code> : Delete the argument of <code>\pcol@makenormalcol</code> because now it is not used too tall pre-environment stuff.	176
<code>\pcol@makenormalcol</code> : Completely redesigned to use <code>\@makecol</code> if pre-environment stuff has bottom floats.	178
<code>\pcol@output@switch</code> : Modify the condition of broadcasting $\kappa_0(\sigma)$ and $\kappa_0(\varepsilon)$ accompanied with <code>\ifpcol@sptext</code> from <code>\ifpcol@sync</code> to $c = 0$ so that the broadcast is made in the first column-switching in column-scanning.	182
<code>\pcol@output@switch</code> : Modify the code structure to let <code>\if@tempswa = true</code> according to the modification of the broadcast of $\kappa_0(\sigma)$ and $\kappa_0(\varepsilon)$	182
<code>\pcol@restartcolumn</code> : Change the code structure to move the insertion of page break penalty for ordinary column-wise footnotes from below the main text to below the footnotes.	183
General: Move commands outside <code>\output</code> routine to the newly introduced section “Commands for Text Coloring” to distinguish macros inside and outside <code>\output</code> routine.	187
<code>\pcol@shrinkcolbyfn</code> : Add the third argument $\langle skip \rangle$ to avoid accidental destruction of <code>\@tempdimb</code> which was modified unconditionally.	192
<code>\pcol@deferredfootins</code> : Fix the bug that the height cap was underestimated by the duplicated subtraction of <code>\skip\footins</code> if the page has already have non-deferred footnotes.	192
<code>\pcol@putfootins</code> : Remove <code>\pcol@output@end</code> from users.	193
<code>\pcol@sync</code> : Add the initialization of <code>\ifpcol@dfloats = false</code> before invoking <code>\pcol@measurecolumn</code>	201
<code>\pcol@sync</code> : Modify the flushing condition of synchronized column switching from $V > \pi^h(p)$ to $\max(V, V - D_T + V_E) > \pi^h(p)$ to avoid page break just below the synchronization point as much as possible.	204
<code>\pcol@flushcolumn</code> : Fix the problem that a flushed column in a non-top page causes overfull due to its high-plus-depth greater than $\pi^h(p)$	204
<code>\pcol@flushcolumn</code> : Add <code>\@maxdepth</code> as the first argument of <code>\pcol@makecol</code>	204

<code>\pcol@measurecolumn</code> : Revise the mechanism to tell <code>\pcol@makeflushedpage</code> and <code>\pcol@output@end</code> that a column in a last page has deferred column-wise floats with newly introduced <code>\ifpcol@dfloats</code>	210
<code>\pcol@makeflushedpage</code> : Revise the mechanism of special care about last page introducing <code>\ifpcol@dfloats</code>	215
<code>\pcol@makeflushedpage</code> : Revise the condition of leaving page-wise floats as ordinary post-environment floats using <code>\if@tempswa</code> with <code>\ifpcol@dfloats</code>	217
<code>\pcol@makeflushedpage</code> : Remove empty column scan for <code>\if@fcolmade</code> because it is now unnecessary thanks to <code>\ifpcol@dfloats</code>	217
<code>\pcol@makeflushedpage</code> : Revise the condition of column-page building and setting of <code>\@colht</code>	217
<code>\pcol@makeflushedpage</code> : page-wise footnotes for the last page followed by pages for deferred column-wise floats are now put by this macro.	218
<code>\pcol@imakeflushedpage</code> : Add <code>\@maxdepth</code> or 0 as the argument of <code>\pcol@makecol</code> to fix the problem that the last page is too large due to <code>\@maxdepth</code> , by the latter.	220
<code>\pcol@imakeflushedpage</code> : Remove the examination of $\kappa_c(\lambda_d)$ for <code>\if@fcolmade</code> because it is made unnecessary now by <code>\ifpcol@dfloats</code>	220
<code>\pcol@output@end</code> : Simplify the case with deferred floats thanks to <code>\ifpcol@dfloats</code> and redesign of <code>\pcol@makeflushedpage</code>	223
<code>\pcol@output@end</code> : Remove <code>\unskip</code> from the operation to let page-wise floats be ordinary ones because <code>\pcol@makeflushedpage</code> does it.	224
<code>\pcol@zparacol</code> : Add operations for the vertical skips at the beginning of a list-like environment.	228
<code>\pcol@sptext</code> : Add globalization of <code>\@svsechd</code> and <code>\@svsec</code>	247
General: Add the section “Commands for Text Coloring” to distinguish macros inside and outside <code>\output</code> routine and describe the latter in this section.	260
<code>\pcol@set@color@push</code> : Change the second argument of <code>\pcol@color@invokeoutput</code> from <code>\hskip\z@</code> to <code>\pcol@fcwhyphenate</code> to make null skip insertion conditional.	262
<code>\coloredwordhyphenated</code> : Introduced to enable null skip insertion before the first word after a coloring command not always but conditionally.	264
<code>\nocoloredwordhyphenated</code> : Introduced to disable null skip insertion before the first word after a coloring command.	264
<code>\pcol@fcwhyphenate</code> : Introduced to enable null skip insertion before the first word after a coloring command not always but conditionally.	264
v1.31	
General: Add passing parameters related sectioning commands beyond <code>\end{paracol}</code> and fix misspells in error messages. (2013/10/10)	1
<code>\pcol@output@end</code> : Add <code>\pcol@getcurrcol</code> for the column specified by <code>\pcol@lastcol</code> to pass <code>\if@nobreak</code> , <code>\if@afterindent</code> and <code>\everypar</code> of the column to post-environment stuff.	226
<code>\pcol@setcw@calcf</code> : Capitalize the first word of the error message for consistency.	239
<code>\pcol@switchcolumn</code> : Add a space before the number of columns in the error message.	246
<code>\pcol@switchenv</code> : Fix the misspell “switching” in the error message.	249
<code>\pcol@twosided</code> : Fix spelling “twosiding” replacing it with “two-siding” in the error message.	258
<code>\pcol@backgroundcolor</code> : Fix the misspelling “colorling” in the error message.	265
<code>\endparacol</code> : Add saving <code>c</code> into <code>\pcol@lastcol</code> to let <code>\pcol@output@end</code> know the column visited last.	268
<code>\pcol@lastcol</code> : Introduced to keep the column visited last to pass its typesetting parameters to post-environment.	268
v1.32-1	
General: Add <code>\globalcounter*</code> to make all counters global. (2015/10/10)	1
General: Add descriptions of <code>\globalcounter*</code>	23
<code>\globalcounter</code> : Modified according to the introduction of <code>\globalcounter*</code>	242
<code>\pcol@globalcounter@s</code> : Added for <code>\globalcounter*</code>	242

<code>\pcol@globalcounter</code> : Renamed from <code>\globalcounter</code> according to the introduction of <code>\globalcounter*</code> .	242
v1.32-2	
General: Fix a memory leak in <code>\pcol@startcolumn</code> . (2015/10/10)	1
<code>\pcol@F@write</code> : Introduced for debugging memory leak problems.	141
<code>\pcol@F</code> : Introduced for debugging memory leak problems.	141
<code>\pcol@FF</code> : Introduced for debugging memory leak problems.	141
<code>\pcol@F@count</code> : Introduced for debugging memory leak problems.	141
<code>\pcol@F@n</code> : Introduced for debugging memory leak problems.	141
<code>\pcol@Fb</code> : Introduced for debugging memory leak problems.	141
<code>\pcol@Fe</code> : Introduced for debugging memory leak problems.	141
<code>\pcol@makecol</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	145
<code>\pcol@cflt</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	148
<code>\pcol@opcol</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	148
<code>\pcol@startpage</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	152
<code>\pcol@outputelt</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	155
<code>\pcol@ioutputelt</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	157
<code>\pcol@ioutputelt</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	157
<code>\pcol@ioutputelt</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	158
<code>\pcol@startcolumn</code> : Fix the memory leak caused by mistakenly preserving $\pi^f(p)$ when $p = p_t$.	164
<code>\pcol@startcolumn</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	164
<code>\pcol@output@start</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	176
<code>\pcol@output@start</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	178
<code>\pcol@output@start</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	178
<code>\pcol@makenormalcol</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	178
<code>\pcol@output@switch</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	180
<code>\pcol@restartcolumn</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	183
<code>\pcol@restartcolumn</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	184
<code>\pcol@flushcolumn</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	204
<code>\pcol@flushcolumn</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	206
<code>\pcol@flushcolumn</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	206
<code>\pcol@makefcolpage</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	208
<code>\pcol@synccolumn</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	212
<code>\pcol@makeflushedpage</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	217
<code>\pcol@makeflushedpage</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	218
<code>\pcol@imakeflushedpage</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	220
<code>\pcol@output@end</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	224
<code>\pcol@output@end</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	225
<code>\pcol@output@end</code> : Add <code>\pcol@Fb/\pcol@Fe</code> pair(s).	226
v1.32-3	
General: Fix a page-wise float problem. (2015/10/10)	1
General: Add footnote to mention the page-wise float problem.	9
General: Add comments about the out-of-order appearance of page-wise floats even with L ^A T _E X-2015/01/10 or later.	60
General: Add the section “Page-wise Float Placement” to discuss the page-wise float problem.	79
<code>\pcol@startpage</code> : Add <code>\f@depth = 0</code> to override <code>\f@depth = 1sp</code> done by <code>\@dblfloatplacement</code> .	152
<code>\pcol@startpage</code> : Modify the code to apply <code>\@sdblcolelt</code> to <code>\@dbldeferlist</code> so as to work with both 2015 (or newer) and 2014 (or older) versions of L ^A T _E X.	152
<code>\pcol@output@start</code> : Add depth clearing of imported deferred floats in case that some of them has 1sp.	174
<code>\pcol@output@clear</code> : Add <code>\f@depth = 0</code> to override <code>\f@depth = 1sp</code> done by <code>\@dblfloatplacement</code> .	214

<code>\pcol@zparacol</code> : Add replacement of <code>\end@dblfloat</code> with <code>\pcol@end@dblfloat</code>	229
<code>\@dbldeferlist</code> : Add top-level definition in case that future L ^A T _E X removes it at all.	233
<code>\pcol@end@dblfloat</code> : Added to go back to old mechanism.	233
v1.33-1	
General: Fix a marginal note problem. (2016/11/19)	1
<code>\pcol@getmpbelt</code> : Fix the bug by which t_k such that $t_k \geq t$ and $t_k - t \geq h$ but $t_k - b_{k-1} < h$ is found.	197
v1.33-2	
General: Non-logical modifications to obey the coding convention, for clarification, etc.. (2016/11/19)	1
<code>\pcol@output</code> : Add a space after <code>\@pcol</code> to obey the coding convention.	143
<code>\pcol@makecol</code> : Move down the <code>\definition</code> of <code>\pcol@currfoot</code> with \perp to place it just before the <code>\ifpcol@scfnote/\fi</code> construct to make it clear how <code>\pcol@currfoot</code> is <code>\defined</code>	145
<code>\pcol@cflt</code> : Add <code>%</code> to the end of the line to open <code>\vbox</code> for <code>\@outputbox</code> to obey the coding convention.	148
<code>\pcol@opcol</code> : Add <code>%</code> to the end of the line to open <code>\vbox</code> for <code>\@currbox</code> to obey the coding convention.	148
<code>\pcol@setpageno</code> : Add <code>\let\@elt\relax</code> before <code>\edef</code> of <code>\reserved@a</code> for the sake of clarity.	149
<code>\pcol@output@switch</code> : Let <code>\dimen@</code> have the height of <code>\pcol@prespan</code> if it is not \perp , or 0 if \perp for the sake of clarity.	180
<code>\pcol@putbackmvl</code> : Add <code>%</code> to the end of the line to open <code>\vbox</code> for <code>\pcol@prespan</code> to obey the coding convention.	186
<code>\pcol@makefcolumn</code> : Remove a space after the <code>\vbox</code> to be assigned to <code>\@currbox</code> to obey the coding convention.	207
<code>\pcol@makefcolpage</code> : Add <code>%</code> to the end of the line to open <code>\vbox</code> to obey the coding convention.	208
<code>\pcol@synccolumn</code> : Add <code>%</code> to the end of the line to open <code>\vbox</code> for <code>\pcol@float</code> to obey the coding convention.	212
<code>\pcol@synccolumn</code> : Add <code>%</code> to the end of the line to open <code>\vbox</code> for <code>\@currbox</code> and two lines for <code>\vboxes</code> in it to obey the coding convention.	213
v1.34	
General: Fix a text coloring problem in non-breakable sequences of vertical items. (2018/05/07)	1
General: Revise the description of §1.6.1 according to the new implementation with <code>\insert</code>	70
General: Revise the description of §1.6.2 according to the new implementation with <code>\insert</code>	71
General: Split the description of <code>\columncolor</code> from §1.6.2 to have new §1.6.3 “Changing Default Column Color” because we have several new issues in the new implementation with <code>\insert</code>	72
General: Revise the description of §1.6.4 according to the new implementation with <code>\insert</code>	73
<code>\pcol@mcid</code> : Change its meaning and operations with it a little bit according to the new text coloring with <code>\insert</code>	130
<code>\pcol@colorstack@samed</code> : Introduced as Γ_s to keep the color stack Γ^c until a column-page of c becomes non-empty.	139
<code>\pcol@tempboxa</code> : Renamed from <code>\pcol@tempbox</code> because its relative <code>\pcol@tempboxb</code> is introduced.	139
<code>\pcol@tempboxa</code> : Add usage in <code>\pcol@scancst</code> and <code>\pcol@iscancst</code>	139
<code>\pcol@tempboxb</code> : Introduced to extract the top of color stack Γ , Γ_r or Γ_s	139
General: Add §3.6 “ <code>\insert</code> Register Set” for <code>\pcol@colorins</code>	140
<code>\pcol@colorins</code> : Introduced to present text-coloring operations to <code>\output</code> synchronously with column-pages.	140
<code>\pcol@ShowBox</code> : Add messaging (VOID) if $\langle b \rangle = \perp$, <code>\vfuzz</code> \leftarrow 0 to ensure overfull, and <code>\mskip</code> of 1 pt if $\langle b \rangle$ ’s height is 0 to ensure overfull too.	140
<code>\pcol@buildcolseprule</code> : Rename <code>\pcol@tempbox</code> as <code>\pcol@tempboxa</code>	158
<code>\pcol@buildcset</code> : Rename <code>\pcol@tempbox</code> as <code>\pcol@tempboxa</code>	158
<code>\pcol@hfil</code> : Rename <code>\pcol@tempbox</code> as <code>\pcol@tempboxa</code>	160

<code>\outputpage</code> : Rename <code>\pcol@tempbox</code> as <code>\pcol@tempboxa</code>	160
<code>\pcol@outputpage@1</code> : Rename <code>\pcol@tempbox</code> as <code>\pcol@tempboxa</code>	162
General: <code>\pcol@op@cpush</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	173
General: <code>\pcol@op@cpop</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	173
General: <code>\pcol@op@cpop</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	173
General: <code>\pcol@op@mcpush</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	174
General: <code>\pcol@op@mcpush@pone</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	174
General: <code>\pcol@op@mcpop</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	174
General: <code>\pcol@op@mcpop@pone</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	174
<code>\pcol@specialoutput</code> : Remove examinations related to $f \in \{\text{cpush}, \text{cpop}, \text{mcpush}, \text{mcpop}\}$. . .	174
<code>\pcol@output@start</code> : Add initialization of $\gamma_0^c = \text{\pcol@columncolor@box} \cdot c$	178
<code>\pcol@putbackmv1</code> : Change nullification of $\Gamma_s = \text{\pcol@colorstack@saved}$ from <code>\gdef</code> to <code>\box-assignment</code> of \perp because it is now a <code>\vbox</code>	186
General: <code>\pcol@output@cpush</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	187
General: <code>\pcol@output@icpush</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	187
General: <code>\pcol@output@mcpush</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	187
General: <code>\pcol@output@imcpush</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	187
General: <code>\pcol@output@cpop</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	187
General: <code>\pcol@reset@color@elt</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	187
General: <code>\pcol@output@mcpop</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	187
General: <code>\pcol@output@mcpop@elt</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	187
General: <code>\pcol@output@cset</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	187
General: <code>\pcol@output@icset</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	187
General: <code>\pcol@return@from@color</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	187
<code>\pcol@clearcolorstack</code> : Completely change its definition according to the new text coloring with <code>\insert</code>	188
General: <code>\pcol@set@color@elt</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	188
<code>\pcol@restorecolorstack</code> : Completely change its definition according to the new text coloring with <code>\insert</code>	189
<code>\pcol@restorecst</code> : Completely change its definition according to the new text coloring with <code>\insert</code>	189
<code>\pcol@scancst</code> : Introduced to implement new text coloring with <code>\insert</code>	189
<code>\pcol@iscancst</code> : Introduced to implement new text coloring with <code>\insert</code>	189
<code>\pcol@savecolorstack</code> : Completely change its definition according to the new text coloring with <code>\insert</code>	191

General: <code>\pcol@colorstack@full</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	191
<code>\pcol@ccuse</code> : Introduced to implement new text coloring with <code>\insert</code>	191
<code>\pcol@ifccdefined</code> : Introduced to implement new text coloring with <code>\insert</code>	191
<code>\pcol@ccxdef</code> : Introduced to implement new text coloring with <code>\insert</code>	191
<code>\pcol@output@end</code> : Remove nullification of γ_0^0 because it is not meaningless now, add release of $\gamma_0^c \neq \perp$ and then nullification of it for all c , move color stack reestablishment down to the loop with c to ensure $\gamma_0^0 = \perp$, and add nullification of Γ	226
<code>\pcol@zparacol</code> : Remove the initializations of <code>\pcol@colorstack</code> and <code>\pcol@colorstack@buf</code> because they no longer exist.	230
General: <code>\pcol@getshadowcc</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	260
<code>\pcol@icolumncolor</code> : Remove the invocations of <code>\pcol@iicolumncolor</code> for <code>\pcol@columncolor@shadow</code> . c because it no longer exists, completely change the operations in the case the target column c is the current one according to the new method with <code>\insert</code> , and add immediate setting of γ_0^c in the case c is not current.	260
<code>\pcol@iicolumncolor</code> : Remove the third argument, change the second argument from a control sequence name to the target column, add a grouping to surround the entire body of the macro, and change the body of γ_0^c so that it only has the color information.	260
<code>\pcol@scancst@shadow</code> : Introduced to rewind or establish the color stack $\hat{\Gamma}^c$	260
<code>\pcol@mcpushlimit</code> : Move down to place it just before the <code>\definition</code> of <code>\pcol@set@color@push</code> being the sole referrer, and change its body from 100 to 1000.	262
<code>\pcol@set@color@push</code> : Completely change its definition according to the new text coloring with <code>\insert</code>	262
<code>\pcol@reset@color@pop</code> : Completely change its definition according to the new text coloring with <code>\insert</code>	263
<code>\pcol@reset@color@mpop</code> : Completely change its definition according to the new text coloring with <code>\insert</code>	263
General: <code>\pcol@color@invokeoutput</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	263
General: <code>\pcol@color@invokeoutput@v</code> is removed according to the change of text coloring from <code>\output</code> to <code>\insert</code>	263
v1.35-1	
General: Fix a page break problem with a tall item in the very first line of the first column. (2018/12/31)	1
<code>\pcol@output@start</code> : Add <code>\interlinepenalty</code> insertion for the first column to avoid vertical overflow due to the first item taller than the column room.	178
v1.35-2	
General: Fix a vertical space problem with <code>trivlist</code> immediately surrounding <code>paracol</code> . (2018/12/31)	1
<code>\pcol@zparacol</code> : Add <code>-\if@inlabel</code> to the condition to perform operations for first <code>\item</code>	228
v1.35-3	
General: Fix a bug in <code>\pcol@addmarginpar</code> and add emulation of <code>\marginnote</code> . (2018/12/31)	1
General: Add description of <code>\marginnote</code>	23
<code>\pcol@addmarginpar</code> : Fix the bug referring to <code>\marbox</code> inappropriately.	194
<code>\pcol@addmarginpar</code> : Add vertical shifting of marginal note to emulate of <code>\marginnote</code>	194
<code>\pcol@zparacol</code> : Add local modifications of <code>\marginpar</code> , <code>\mn@marginnote</code> and <code>\xympar</code> for the emulation of <code>\marginnote</code>	231
General: Add the section “Commands for Marginal Notes” to describe newly introduced macros for the emulation of <code>\marginnote</code>	257
<code>\marginpar</code> : Locally modified in <code>\pcol@zparacol</code> for the emulation of <code>\marginnote</code>	257
<code>\pcol@marginpar</code> : Introduced as the in- <code>paracol</code> version of <code>\marginpar</code> for the emulation of <code>\marginnote</code>	257

\pcol@marginpar: Introduced to keep the original version of \marginpar for the emulation of \marginnote.	257
\@mn@marginnote: Locally modified in \pcol@zparacol for the emulation of \marginnote.	257
\pcol@marginnote: Introduced as the in-paracol version of \@mn@marginnote for the emulation of \marginnote.	257
\pcol@mn@warning: Introduced to put a warning message to show \marginnote is emulated.	257
\@xympar: Locally modified in \pcol@zparacol for the emulation of \marginnote.	257
\pcol@xympar: Introduced as the in-paracol version of \@xympar for the emulation of \marginnote.	257
\pcol@xympar: Introduced to keep the original version of \@xympar for the emulation of \marginnote.	257
\pcol@mparoffset: Introduced to have the vertical offset for the emulation of \marginnote.	257
v1.35-4	
General: Add \belowfootnoteskip for the additional space below the non-merged pre-environment footnotes. (2018/12/31)	1
General: Add description of \belowfootnoteskip.	15
General: Add description of \belowfootnoteskip.	25
\belowfootnoteskip: Introduced to specify the additional space below the non-merged pre-environment footnotes.	138
\pcol@output@start: Add \belowfootnoteskip to H_f being the space for the non-merged pre-environment footnotes.	174
\pcol@combinefootins: Add the insertion of vertical skip \belowfootnoteskip.	193
v1.35-5	
General: Add \definecolumnpreamble. (2018/12/31)	1
General: Add description of \definecolumnpreamble.	18
\pcol@zparacol: Add the invocation of \pcol@colpream-0.	232
\pcol@switchcol: Add the invocation of \pcol@colpream·c.	247
\definecolumnpreamble: Introduced to define a column preamble.	250
v1.35-6	
General: Add error check if paracol environment is not in outer par mode and is with ordinary two-column typesetting. (2018/12/31)	1
\pcol@zparacol: Add error check with \ifinner and \if@twocolumn.	228
v1.36-1	
\pcol@switchcol: Add check to not add zeroth preamble in \endparacol.	247
v1.36-2	
General: Rename \footnotelayout to \footnoteplacement.	24
\pcol@zparacol: Add new API inactivation for \footnoteplacement and a check before deactivating \footnotelayout.	231
\footnoteplacement: Rename \footnotelayout to \footnoteplacement for compatibility with footmisc package.	253
v1.36-3	
\pcol@zparacol: Check for \@footnotetext being renamed to \H@@footnotetext.	231