The parcolumns package*

Jonathan Sauer
jonathan.sauer@gmx.de

翻译
virhuiai@qq.com

2004/11/25

摘要

本文件描述的是 parcolumns 宏包，该宏包提供了一个环境，可以将文本以两列或多列的方式并列排版。

This file describes the parcolumns package that provides an environment for typesetting text in two or more columns columns in parallel.

目录

*This document corresponds to parcolumns.sty v1.2, dated 2004/11/25.

1 Introduction
介绍

Sometimes it is necessary to typeset text in two or more columns in parallel, i.e. when typesetting a text in its original language and in its translation(s). This package provides the `parcolumns` environment for typesetting text in several columns, where the text of all columns is aligned vertically.

有时需要在多列中并列地排版文本，例如当排版一个原文和其翻译时。本宏包提供了 `parcolumns` 环境，用于在多列中并列地排版文本，所有列的文本在垂直方向上对齐。

Text in the `parcolums` environment is typeset in chunks, where the chunks of one row are aligned vertically. A chunk consists of one or more paragraphs of text, including TEX macros.

在 `parcolums` 环境中，文本是以块的形式排版的，一行中的块在垂直方向上对齐。块包括一个或多个文本段落，包括 TEX 宏。

2 Usage
用法

`parcolumns` Usage: `parcolumns [`⟨*options*⟩`] {`⟨*number of colums*⟩`}`.

Inside the `parcolumns` environment text can be typeset in two or more columns in parallel via the `\parchunk` macro. Normal text can also be included.

在 `parcolumns` 环境中，可以使用 `\parchunk`[1]宏在两个或多个列中并列地排版文本。也可以包括普通文本。

The `parcolumns` environment takes an optional parameter that specifies the

[1]译注：似乎没有 `\parchunk` 这个命令？

options for the environment using the keyval and processkv packages. The following options exist:

parcolumns 环境接受一个可选参数，该参数指定环境的选项，使用 keyval 和 processkv 宏包。存在以下选项：

colwidths Sets the widths of the columns. The widths are specified as key-values, ⟨*columnnumber*⟩=⟨*width*⟩. Columns start at '1'. Note that in order to be ignored by the keyval package, the complete value has to be surrounded by braces, i.e. '`colwidths={1=2cm,3=5cm,4=2cm}`' to set the first column to a width of 2 cm, the third to 5 cm and the fourth to 2 cm. (the second column is calculated)

设置列的宽度，宽度以键值对的形式指定，⟨*columnnumber*⟩=⟨*width*⟩，其中列从 1 开始编号。请注意，为了避免被 keyval 包解析，必须用大括号将完整值括起来，例如 '`colwidths={1=2cm,3=5cm,4=2cm}`' 将第一列设置为 2cm 宽，第三列设置为 5cm 宽，第四列设置为 2cm 宽。(第二列的宽度将被计算)。

第 1 列设置为 2cm 宽	第 2 列宽度将被计算	第 3 列设置为 5cm 宽	第 4 列设置为 2cm 宽

Columns not set this way will be distributed evenly across the remaining horizontal space of the page.

没有设置宽度的列将均匀分布在页面的剩余水平空间中。

第 1 列均匀分布	第 2 列均匀分布	第 3 列均匀分布	第 4 列均匀分布 AAA

distance Sets the distance between two columns. If omitted set to `2em`.

设置两列之间的距离。如果省略，则设置为 `2em`。

rulebetween (Flag)[2] Typeset a vertical bar between two columns.

(标志)[3] 在两列之间绘制垂直条形线。

[2]Meaning that if you just say `rulebetween`, you set the flag, as well as when you say `rulebetween=true`. Saying `rulebetween=false` clears the flag, even though this does not make much sense, as it is the default.

[3]这意味着，如果只写 `rulebetween`，则设置标志，与写 `rulebetween=true` 相同。说 `rulebetween=false`

`false` if omitted.

如果省略，则设置为 `false`。

nofirstindent (Flag) Suppress the indentation of the first paragraph in the environment.

（标志）取消环境中第一段的缩进。

`false` if omitted.

如果省略，则设置为 `false`。

sloppy (Flag) Typeset text in the columns in a more sloppy way, preventing overfull hboxes at the cost of larger interword spacing.

（标志）以更松散的方式在列中排版文本，以防止出现 overfull hbox，但代价是更大的单词间距。

`false` if omitted.

如果省略，则设置为 `false`。

sloppyspaces (Flag) Makes spaces in the columns stretchable, preventing overfull hboxes at the cost of even larger interword spacing that `sloppy`.

（标志）使列中的空格具有可伸缩性，以防止出现 overfull hbox，但代价是比`sloppy` 更大的单词间距。

`false` if omitted.

如果省略，则设置为 `false`。

`\colchunk` Usage: `\colchunk [`⟨*column*⟩`] {`⟨*chunk*⟩`}`.

The macro `\colchunk` sets a chunk of text for the next column. You don't have to set chunks for all columns; the colums not set remain empty. However, the columns are filled from left to right, so if a column inbetween should remain empty, you must say `\colchunk{}` or specify the column to set using the optional parameter (columns start at 1): `\colchunk[2]{…}` sets the text of the second column. Following calls to `\colchunk` without optional parameter fill the third, fourth et cetera column.

会清除该标志，尽管这没有多大意义，因为它是默认值。

宏 \colchunk 为下一列设置一个文本块。不必为所有列设置块；未设置块的列将保持为空。但是，列从左到右填充，因此如果中间的某一列应保持为空，则必须用 \colchunk{} 或使用可选参数指定要设置的列（列从 1 开始）：\colchunk[2]{…} 设置第二列的文本。接下来的对\colchunk 的调用不带可选参数则将填充第三、第四等列。

After a call to \colplacechunks, the column number is reset to one.

在调用\colplacechunks 后，列号重置为 1。

\colplacechunks Usage: \colplacechunks.

The macro \colplacechunks places the chunks added with \colchunk on the page. If there are no chunks to place, it does nothing.

宏\colplacechunks 将使用\colchunk 添加的文本块放置在页面上。如果没有要放置的块，则不执行任何操作。

3 Examples
示例

Two columns, option rulebetween=true (which is the same as just saying rulebetween):

两列，选项rulebetween=true（与仅使用rulebetween 相同）：

```
\addvspace{\baselineskip}
\begin{parcolumns}[rulebetween=true]{2}
\colchunk{Erwarten Sie von mir, dass ich rede?...}
\colchunk{Do you expect me to talk? -- No,...}
\colplacechunks
\colchunk{Erwarten Sie von mir, dass ich rede?...}
\colchunk{Do you expect me to talk? -- No,...}
\end{parcolumns}
```

Erwarten Sie von mir, dass ich rede? – Nein, Mister Bond, ich erwarte von Ihnen, dass Sie sterben!

Do you expect me to talk? – No, Mister Bond, I expect you to die!

Erwarten Sie von mir, dass ich rede? – Nein, Mister Bond, ich erwarte von Ihnen, dass Sie sterben!

Do you expect me to talk? – No, Mister Bond, I expect you to die!

As the german text is slightly longer, let's make the left column alittle bit larger using `colwidths={1=.55\linewidth}`:

由于德文文本略长，让我们使用`colwidths={1=.55\linewidth}` 将左侧列稍微放大：

```
\begin{parcolumns}[rulebetween=true,colwidths={1=.55\linewidth}]{2}
```

Erwarten Sie von mir, dass ich rede? – Nein, Mister Bond, ich erwarte von Ihnen, dass Sie sterben!
Erwarten Sie von mir, dass ich rede? – Nein, Mister Bond, ich erwarte von Ihnen, dass Sie sterben!

Do you expect me to talk? – No, Mister Bond, I expect you to die!
Do you expect me to talk? – No, Mister Bond, I expect you to die!

Three columns, option `nofirstindent=true`:

三列，选项`nofirstindent=true`：

```
\begin{parcolumns}[nofirstindent]{3}
```

This is just a short english text, just long enough to fill a few lines.

Dies ist nur ein kurzer deutscher Text, gerade lang genug, um ein paar Zeilen zu fuellen.

This is another short english text, as my french is not that good anymore.

There was an overfull `\hbox` in the previous text. Let's try that again with the option `sloppy`:

上一个文本中有一个 overfull 的`\hbox`。让我们尝试使用选项`sloppy`：

```
\begin{parcolumns}[nofirstindent,sloppy]{3}
```

This is just a short english text, just long

Dies ist nur ein kurzer deutscher Text, gerade

This is another short english text, as my

enough to fill a few lines.

lang genug, um ein paar Zeilen zu fuellen.

french is not that good anymore.

No overfull hbox this time, but the spacing in the first column is not optimal. You just have to pick what's better. Or you could try the interword spacing provided by the soul package.

这次没有出现过满的 hbox，但第一列中的间距不太理想。你只需选择更好的选项。或者你可以尝试使用 soul 包提供的字间距调整。

Maybe we do not want to fill the first column, but do not want to type `\colchunk{}` either. Then we can say `\colchunk[2]` to directly start with the second column (note that we are using the option `sloppy`, too):

也许我们不想填充第一列，但也不想输入 `\colchunk{}`。这时我们可以使用 `\colchunk[2]` 直接从第二列开始（注意我们也使用了选项 `sloppy`）：

```
\begin{parcolumns}[nofirstindent,sloppy]{3}
\colchunk[2]{This is just a ...}
\colchunk{Dies ist nur ein...}
\colplacechunks
\end{parcolumns}
```

This is just a short english text, just long enough to fill a few lines.

Dies ist nur ein kurzer deutscher Text, gerade lang genug, um ein paar Zeilen zu fuellen.

Note that `parcolumns` does not insert vertical space before or after the environment! In these examples, the space has manually been added with `\addvspace`.

请注意，`parcolumns` 不会在环境前后插入垂直空间！在这些示例中，空间是手动使用 `\addvspace` 添加的。

4 Notes
注意

- The columns will always fill the complete width of the page by stretching or shrinking the space between the columns. That means that if the total width of all columns is set to about half the page width, the space between the columns will take up the rest, ignoring whatever was set with the key `distance`.[4]

  列始终会通过拉伸或缩小列之间的空间来填充页面的整个宽度。这意味着，如果所有列的总宽度设置为大约页面宽度的一半，则列之间的空间将占用其余空间，而忽略使用关键字 `distance` 设定的任何距离。[5]

- Footnotes are not set in columns.

  脚注不会设置在列中。

- parcolumns does not work very well with displayed formula. The best way to typeset a `displaymath` et.al. environment is to include it in separate `\colchunk` commands, i.e. (assuming two columns and the same formula in both):

  parcolumns 与 displayed 公式的配合效果不是很好。排版 `displaymath` 环境等最佳方式是在单独的 `\colchunk` 命令中包含它们，即（假设有两列，且两列中的公式相同）:

```
... some text placed using \colchunk ...
\colchunk{ % Left column
\begin{displaymath}
x^2 + y^2 = z^2
\end{displaymath}
}
\colchunk{ % Right column
\begin{displaymath}
x^2 + y^2 = z^2
\end{displaymath}
}
\colplacechunks
... some more text placed using \colchunk ...
```

[4] The key `distance` is only used for calculating the width of the columns and is ignored afterwards.

[5] 关键字 `distance` 仅用于计算列的宽度，在之后被忽略。

5 Implementation
实现

5.1 Allocation
分配

The current column (starting with one):

当前列（从 1 开始）：

```
\newcount\pc@columnctr
```

The total number of columns in the current `parcolums` environment.

当前`parcolumns` 环境中的列总数：

```
\newcount\pc@columncount
```

Place a vertical rule between two columns?

在两列之间放置垂直线？

```
\newif\ifpc@rulebetween
```

Stores `\everypar` for use in `\parparagraphs`.

存储 `\everypar` 以在 `\parparagraphs` 中使用。

```
\newtoks\pc@everypar
```

Note that additional allocations are performed later on-demand in `\pc@alloccolumns`.

请注意，稍后在 `\pc@alloccolumns` 中会根据需要执行其他分配。

5.2 keyvalue keys

`\pc@boolkey` Sets an `if`-condition in `{`⟨*#1*⟩`}` to the value passed as `{`⟨*#2*⟩`}`. If `#2` is `false`, set `if#1` to `false`, else (any other value) to `true`.

将 `{`⟨*#1*⟩`}` 中的 `if`-条件设置为传递的值 `{`⟨*#2*⟩`}`。如果 `\#2` 是 `false`，则将 `if\#1` 设置为 `false`，否则（任何其他值）设置为 `true`。

Usage: `\pc@boolkey {`⟨*ifcondition*⟩`} {`⟨*value*⟩`}`

```
\def\pc@boolkey#1#2{%
  \edef\@tempa{#2}%
  \edef\@tempb{false}%
  \ifx\@tempa\@tempb%
    \csname #1false\endcsname%
  \else%
    \csname #1true\endcsname%
  \fi%
}
```

Define the keys for the options of `parcolumns`.

定义 `parcolumns` 的选项的键。

```
\define@key{parcolumns}{distance}{%
  \@tempdimc#1\relax%
}
\define@key{parcolumns}{rulebetween}[true]{%
  \pc@boolkey{pc@rulebetween}{#1}%
}
\define@key{parcolumns}{nofirstindent}[true]{%
  \pc@boolkey{@tempswa}{#1}%
```

If the indentation of the first line of the first paragraph should be suppressed, change `\pc@everypar` accordingly. The token register is reset after the placing of the first row of chunks in `\colplacechunks`.

如果需要取消第一个段落的第一行缩进，可以相应地更改`\pc@everypar`。在将第一行块放置后，标记寄存器将被重置。

What exactly are we doing? First we assign box 0 the contents of `\lastbox`. At the beginning of a paragraph `\lastbox` contains the glue inserted for the indentation of the first line. By assigning this box to box 0, we remove the indentation. We do this in a group as not to change the contents of box 0.

我们到底在做什么？首先，我们将盒子 0 赋值为`\lastbox` 的内容。在段落的开头，`\lastbox` 包含为第一行缩进插入的粘连。通过将此盒子赋值给盒子 0，我们删除了缩进。我们在一个组中执行此操作，以不更改盒子 0 的内容。

Afterwards we set `\everypar` to nothing. This is necessary because we only want to suppress the indentation for the first paragraph of the `parcolumns`

environment, not every paragraph. We set \everypar and not \pc@everypar to nothing, because \pc@everypar is only a temporary storage that will be assigned to \everypar when used.

然后，我们将\everypar 设置为空。这是必要的，因为我们只想在parcolumns 环境的第一个段落中取消缩进，而不是每个段落。我们将\everypar 设置为空，而不是\pc@everypar，因为\pc@everypar 只是一个临时存储，当使用时将被赋值给\everypar。

```
  \if@tempswa\pc@everypar{{\setbox\z@\lastbox}\everypar{}}\fi%
}
\define@key{parcolumns}{sloppy}[true]{%
  \pc@boolkey{@tempswa}{#1}%
```

If sloppy typesetting is asked for, we set \hbadness and \tolerance to 10000, so that TEX breaks lines whenever possible, even if this means high interword spacing.

如果要求松散排版，则将\hbadness 和\tolerance 设置为 10000，以便在可能的情况下断行，即使这意味着高间距。

```
  \if@tempswa%
    \hbadness\@M%
    \tolerance\@M%
  \fi%
}
\define@key{parcolumns}{sloppyspaces}[true]{%
  \pc@boolkey{@tempswa}{#1}%
```

If sloppy spaces are asked for, we make the space stretchable:

如果需要松散的空间，我们会让空间可伸缩：

```
  \if@tempswa%
    \spaceskip.3333em\@plus1em %
  \fi%
}
```

We save the key-value-list containing widths of the columns in \toks@ for later use in \pc@setcolumnwidths.

将包含列宽的键值列表保存在\toks@ 中，以供以后在\pc@setcolumnwidths

中使用。

```
\define@key{parcolumns}{colwidths}{%
  \toks@{#1}%
}
```

5.3 Main environments and macros
主要环境和宏命令

`parcolumns` Environment for a number of columns of text set in parallel (see section 2 on page 2 for the possible options)

平行排列文本的环境（有关可能的选项，请参见第 2页的第 2节）

Usage: `\begin{parcolumns}` [⟨*options*⟩] `{`⟨*number of columns*⟩`}` ... `\end{parcolums}`

```
\newenvironment{parcolumns}[2][]{%
  \pc@rulebetweenfalse%
```

`\if@tempswa` is `true`, if the indentation of the first line should be suppressed, otherwise `false`. Default is `false`:

如果第一行的缩进应该被抑制，则 `\if@tempswa` 为 true，否则为 false。默认为 false：

```
  \@tempswafalse%
```

`\@tempdimc` contains the space between two columns. Default is `2em`:

`\@tempdimc` 包含两列之间的间距。默认值为 `2em`：

```
  \@tempdimc2em\relax%
```

We set the options:

我们设置选项：

```
  \toks@{}%
  \setkeys{parcolumns}{#1}%
```

We store the total number of columns and reset the counter for the current column:

我们存储总列数并重置当前列的计数器：

```
\pc@columncount#2 %
\pc@columnctr\z@%
```

We allocate the columns and set their widths:

我们分配列并设置它们的宽度：

```
\pc@alloccolumns%
\pc@setcolumnwidths%
```

We switch to vertical mode:

我们切换到垂直模式：

```
\endgraf%
```

As we are changing `\everypar`, we need to make sure that the most important flag is reset, which normally happens in the `\everypar` of a `\section`-command: `\if@nobreak`. (otherwise the next section is typeset directly after the text of the previous section instead of leaving some space)[6]

由于我们正在改变 `\everypar`，因此我们需要确保最重要的标志被重置，这通常发生在 `\section` 命令的 `\everypar` 中：`\if@nobreak`。（否则，下一个部分将直接在前一个部分的文本后面排版，而不是留下一些空间）[7]

```
\@nobreakfalse%
```

We reset `\everypar`, as it is of no use for us and can only screw up things badly:

我们重置 `\everypar`，因为它对我们没有用处，只会使事情变得混乱：

```
\global\everypar{}%
}{%
```

At the end of the `parcolumns`-environment … just in case, we place the last chunks, if not already done so:

[6] I discovered this the hard way, spending one or two hours wondering why the spacing between two sections was much too small.

[7] 我以艰难的方式发现了这一点，花了一两个小时想知道为什么两个部分之间的间距太小了。

在 parcolumns 环境结束时 … 以防万一，如果尚未这样做，则放置最后的块：

```
\colplacechunks%
\endgraf%
```

We reset \clubpenalty globally to its normal value (the \global makes sure that if \everypar should have reset \clubpenalty, it is reset now).

我们将全局 \clubpenalty 重置为其正常值（\global 确保如果 \everypar 应该重置 \clubpenalty，它现在被重置）。

```
\global\clubpenalty\@clubpenalty%
```

We suppress the indentation of the next paragraph immediately following the environment:

我们立即抑制环境后面的下一个段落的缩进：

```
\@doendpe%
}
```

\colchunk Sets the text for the next chunk of the next column.

为下一列的下一块设置文本。

Usage: \colchunk [⟨*column*⟩] {⟨*chunk*⟩}. (note that the { and } are *not* optional, even if the chunk consists of only one token!)

使用：\colchunk [⟨*column*⟩] {⟨*chunk*⟩}。（请注意，即使该块仅包含一个标记，{和} 也是不可选的！）

We need two macros for handling the optional parameter ⟨*column*⟩, \colchunk and \colchunk@. First we define \colchunk:

我们需要两个宏来处理可选参数⟨*column*⟩，即\colchunk 和\colchunk@。首先我们定义\colchunk：

```
\newcommand{\colchunk}{\@testopt\colchunk@{}}%
```

\colchunk@ What are we doing now? By suffixing the last (optional) parameter by #, we tell TEX that the last parameter of \colchunk is to be delimited by a right brace {, or TEX will complain with a more comprehensible message that

without the #: 'Use of \colchunk doesn't match its definition.' instead of 'Missing { inserted.'

现在我们在最后（可选）参数后面加上#，这告诉 TEX，\colchunk 的最后一个参数将由右括号{界定，否则 TEX 将抱怨更易理解的消息，而不是没有#: '使用\colchunk 不符合它的定义'，而不是 '缺少{插入'。

```
\long\def\colchunk@[#1]#{%
```

We check if the optional parameter is not empty. Then we use it as the column number, otherwise we simply pick the next column:

我们检查可选参数是否为空。如果不为空，则将其用作列号，否则我们仅选择下一列：

```
  \ifx\\#1\\%
    \advance\pc@columnctr\@ne%
  \else%
    \pc@columnctr#1\relax%
  \fi%
```

If we try to add a column past the last one, we display an error message:

如果我们试图添加一个超出最后一列的列，我们会显示错误消息：

```
  \ifnum\pc@columnctr>\pc@columncount%
    \PackageError{parcolumns}{The column \number\pc@columnctr\space%
      is too large}{Only \number\pc@columncount\space columns are%
      \space allowed.}
```

As we cannot simply skip the chunk (we could gobble what follows after the macro, but this would require a lot of jumping, and just for handling a small mistake …), we simply set the last column:

由于我们不能简单地跳过该块（我们可以吞掉宏后面的内容，但这需要很多跳转，仅用于处理小错误…），我们只需将最后一列设置为当前列：

```
    \pc@columnctr\pc@columncount%
  \fi%
```

We zero the \clubpenalty that could have been changed, i.e. by \everypar of a \section-command: \section changes the clubpenalty to prevent a break between the first two lines of the paragraph following immediately after the

section; as we split off line after line when typesetting the two columns, this would make splitting of a single line impossible (`\vsplit` uses the same logic as page-breaking), thus resulting in a lot of overfull vboxes and weird spacing inbetween, as *two* lines would be split off.

我们重置`\clubpenalty`以防止它被改变，例如通过`\everypar`的`\section`命令：`\section`会改变 clubpenalty 来防止段落的前两行之间出现分隔；由于我们在排版两列时会一行一行地分割，这会使单行分割不可能（`\vsplit`使用与页面分割相同的逻辑），因此会导致大量的溢出 vbox 和诡异的字间距，因为两行会被分割。

We do this every time we add chunks just in case some macro in the chunks has changed `\clubpenalty`. Note that we cannot prevent a macro in the text of the chunk to change the `\clubpenalty`.[8] But if it is changed, at least it will not stay changed (though the typeset columns will still look bad).

我们每次添加块时都会这样做，以防某些块中的宏改变了`\clubpenalty`。注意，我们不能防止块文本中的宏改变`\clubpenalty`。[9]但是，如果它被改变了，至少它不会保持改变（尽管排版的列看起来仍然很糟糕）。

```
72  \clubpenalty\z@%
```

The same goes for the other penalties TEX will insert between two lines, as we absolutely, positively *must* be able to break between any two lines:[10]

我们也对 TEX 在两行之间插入的其他惩罚都这样做，因为我们绝对，绝对必须能够在任何两行之间断行：[11]。

```
73  \interlinepenalty\z@%
74  \displaywidowpenalty\z@%
75  \widowpenalty\z@%
76  \brokenpenalty\z@%
```

We set `\everypar` to our self-defined `\pc@everypar` to suppress the indentation of the first paragraph if so desired:

[8] Well we could, by redefining the clubpenalty to be a macro that simply throws its parameter away, and using some tricks to make this macro behave like a register, therefore completely ignoring any change of the clubpenalty.

[9] 好吧，我们可以通过重新定义 clubpenalty 来使它什么也不做，并使用一些技巧使这个宏表现得像 register 一样，因此完全忽略 clubpenalty 的任何改变。

[10] C.f. chapter 14 of the TEXbook

[11] 参见《TEX》书第 14 章

我们将\everypar 设置为自定义的\pc@everypar 以在需要时抑制段落的缩进：

```
\everypar\expandafter{\the\pc@everypar}%
```

After the next assignment, insert the width of the column:

在下一个赋值之后，插入列的宽度：

```
\afterassignment\pc@setcolumnwidth%
```

We typeset the chunk's text into the box \pc@column@⟨*column counter*⟩. The text of the chunk follows the macro, meaning that the last line looks like this: \vbox{⟨*chunk text*⟩}.

我们将块的文本排版到\pc@column@⟨*column counter*⟩。块的文本在宏之后，这意味着最后一行看起来像这样：\vbox{⟨*chunk text*⟩}。

But what about the width of the box? \hsize must be set in the vbox in order to make it the correct width! We achieve this using the \afterassignment above: After the assignment of the chunk's text to the box, we are inside the box, therefore the contents of \pc@setcolumnwidth is inserted at the very beginning of the vbox, setting the correct width.

但是 box 的宽度呢？必须在 vbox 中设置\hsize 才能使它具有正确的宽度！我们通过上面的\afterassignment 来实现：在将块的文本分配给 box 之后，我们就在 box 内了，因此\pc@setcolumnwidth 中的内容会插入到 vbox 的最前面，从而设置正确的宽度。

Why don't we simply make \colchunk take one parameter that contains the text of the chunk? Because in that case, macros that change catcodes like \verb would be prohibited, which would restrict this package somewhat. Also it would be slower and would use more memory.

为什么我们不简单地使\colchunk 接受一个包含块文本的参数？因为在这种情况下，像\verb 这样改变 catcodes 的宏被禁止，这会限制这个软件包。此外，它运行起来会更慢，而且会使用更多的内存。

```
  \expandafter\setbox\csname pc@column@\number\pc@columnctr\endcsname%
    \vbox%
}
```

\colplacechunks Places all chunks set with \colchunk.

放置使用\colchunk 设置的所有块。

```
\newcommand{\colplacechunks}{%
```

If there are any chunks to place:

如果有任何要放置的块：

```
  \ifnum\pc@columnctr>\z@%
```

We place them:

我们放置它们：

```
    \pc@placeboxes%
```

We reset the column counter:

我们重置列计数器：

```
    \pc@columnctr\z@%
```

We clear \pc@everypar, because we only want to suppress the indentation of the first paragraph of each column at the very top of the parcolumns environment, not of the first paragraph of every call to \colchunk.

我们清除 \pc@everypar，因为我们只想在 parcolumns 环境的顶部的每个列的第一段落抑制缩进，而不是每个 \colchunk 调用的第一段落。

```
    \pc@everypar{}%
  \fi%
}
```

5.4 Internal environments and macros
内部环境和宏

\pc@placeboxes Places the prepared boxes (containing the chunks) on the page.

放置已准备好的盒子（包含块）到页面上。

```
\def\pc@placeboxes{%
```

We assume we don't have to perform another line (`\@tempa` contains what we have to to after we are finished with this macro). The assignment is global because later on when changing `\@tempa` we are in a group:

假设我们不必再执行另一行（`@tempa` 包含我们完成此宏后要执行的内容）。这个赋值是全局的，因为当我们更改 `@tempa` 时，我们处于一个组内：

```
\global\let\@tempa\relax%
\count@\z@%
```

We create a hbox. Inside a hbox, vboxes are put horizontally next to each other and are aligned on their baseline:

我们创建一个 hbox。在 hbox 中，vbox 会水平放置在一起并在它们的基线上对齐：

```
\hb@xt@\linewidth{%
```

Before doing any work, we change a few parameters:

在做任何工作之前，我们更改一些参数：

We prevent warnings of overfull and underfull vboxes as they can happen, but we do not really care (happens when we `\vsplit` the top off the chunks, this is okay):

我们防止过满和过空的 vbox 警告，因为它们可能会发生，但我们并不真的关心（当我们从块的顶部使用 `\vsplit` 时发生，这是可以接受的）：

```
    \vfuzz30ex %
    \vbadness\@M%
```

We prevent vertical glue to be insert when `vsplit`ting a vbox (otherwise it screws up the spacing).

我们在 `vbox` 分割时防止插入垂直粘合剂（否则会弄乱间距）。

```
    \splittopskip\z@skip%
```

Now we loop over all the prepared boxes. We can use `\loop` here as we are in a group (begun by `\hbox`). Otherwise we would have to open a group manually or loop manually, as to not screw up a `\loop` outside the macro:

现在我们循环遍历所有准备好的盒子。我们可以在此使用 \loop，因为我们在一个组中（由 \hbox 开始）。否则，我们需要手动打开一个组或手动循环，以避免破坏宏外部的 \loop：

```
\loop\ifnum\count@<\pc@columncount%
  \advance\count@\@ne%
```

If the box \pc@column@⟨*counter*⟩ is empty, we simply insert a hskip the width of the box (saved in \pc@column@width@⟨*counter*⟩):

如果盒子 \pc@column@⟨*counter*⟩ 为空，我们只需插入一个宽度为该盒子中保存的宽度（在 \pc@column@width@⟨*counter*⟩ 中）的 hskip：

```
  \expandafter\ifvoid\csname pc@column@\number\count@%
    \endcsname%
    \hskip\csname pc@column@width@\number\count@\endcsname%
  \else%
```

Otherwise we \vsplit the first line of the box (at the same time removing it from \pc@column@⟨*counter*⟩) and save it in \@tempboxa. Then, we strip this resulting box of its enclosing vbox and put it into another vbox, which we put into the hbox begun a few lines ago.

否则，我们将盒子 \pc@column@⟨*counter*⟩ 的第一行分割（同时从 \pc@column@⟨*counter*⟩ 中删除它）并将其保存在 @tempboxa 中。然后，我们剥离这个结果盒子的包含 vbox 并将其放入另一个 vbox 中，然后将其放入前面几行开始的 hbox 中。

Why is this so complicated? To ensure that proper vertical glue is inserted (otherwise the spacing between the lines would be wrong).

为什么这么复杂？为了确保插入适当的垂直粘合剂（否则行之间的间距将不正确）。

```
    \expandafter\setbox\expandafter\@tempboxa\expandafter%
      \vsplit\csname pc@column@\number\count@\endcsname%
        to \dp\strutbox%
    \vbox{\unvbox\@tempboxa}%
  \fi%
```

If the remaining box is not empty, we have to perform at least another line:

如果剩余的盒子不为空，我们就必须执行至少另一行：

```
      \expandafter\ifvoid\csname pc@column@\number\count@%
        \endcsname\else%
        \global\let\@tempa\pc@placeboxes%
      \fi%
```

If this is not the last column, we put a strut into the hbox to ensure propery vertical spacing:

如果这不是最后一列，我们会将一个支撑线放入 hbox 中，以确保垂直间距正确：

```
      \ifnum\count@<\pc@columncount%
        \strut%
```

If a vertical line should be placed between two columns, we insert it now, centering it between two \hfills. Otherwise, we simply insert a \hfill that stretches as much as possible, pushing the right column to the right margin. (see also section 4 on page 8)

如果应在两列之间放置垂直线，我们现在插入它，让它居中于两个 \hfill 之间。否则，我们只需插入一个能够尽可能地拉伸的 \hfill，将右列推到右边缘（详见第 4 节，第 8 页）：

```
        \hfill%
        \ifpc@rulebetween%
          \vrule%
          \hfill%
        \fi%
      \fi%
    \repeat%
  }%
```

If necessary, we perform another line:

如果必要，我们执行另一行：

```
  \@tempa%
}
```

\pc@alloccolumns Allocates a number of \boxes, named \pc@column@1, \pc@column@2 et cetera, if not already allocated.

如果尚未分配名为 \pc@column@1、\pc@column@2 等的一些 \box，则会分配它们。

Also allocates a number of \dimens, named \pc@column@width@1 et cetera.

还会分配一些名为 \pc@column@width@1 等的 \dimen。

If the \boxes and \dimens are already allocated, they are cleared (\boxes) or set to zero (\dimens).

如果 \box 和 \dimen 已经分配，它们将被清除（\box）或设为零（\dimen）。

```
\def\pc@alloccolumns{%
  \count@\z@%
  \loop\ifnum\count@<\pc@columncount%
    \advance\count@\@ne%
    \@ifundefined{pc@column@\number\count@}{%
      \expandafter\newbox\csname pc@column@\number\count@%
        \endcsname%
      \expandafter\newdimen\csname pc@column@width@\number%
        \count@\endcsname%
    }{%
      \setbox0\box\csname pc@column@\number\count@\endcsname%
      \csname pc@column@width@\number\count@\endcsname\z@%
    }%
  \repeat%
}
```

\pc@setcolumnwidths Sets the widths of all columns. The defined widths have been temporarily stored in \toks@. \@tempdimc contains the space between two columns.

设置所有列的宽度。定义的宽度已经被临时存储在 \toks@ 中。@tempdimc 包含两个列之间的间距。

```
\def\pc@setcolumnwidths{%
  \expandafter\processkeyvalues\expandafter{\the\toks@}%
    \pc@setsinglecolwidth%
```

\@tempdima will contain the total width of all columns that have a known width (read: have had their width set via the parameter colwidths)

设置所有列的宽度。定义的宽度已经被临时存储在 \toks@ 中。@tempdimc 包

含两个列之间的间距。

```
\@tempdima\z@%
```

`\@tempcnta` will contain the count of columns that an unknown width (read: have had their width *not* set via the parameter `colwidths`, thus now having a width of zero points, as the widths of all columns have been reset in `\pc@alloccolumns`):

`@tempcnta` 将包含未知宽度的列的数量（即：通过参数 `colwidths` 未设置宽度，因此宽度现在为零点，因为在 `\pc@alloccolumns` 中重置了所有列的宽度）。

```
\@tempcnta\z@%
```

We calculate the total width of all columns known. We count how many columns have an unknown width:

我们计算所有已知列的总宽度。我们计算有多少未知宽度的列：

```
\count@\z@%
\loop\ifnum\count@<\pc@columncount%
  \advance\count@\@ne%
  \@tempdimb\csname pc@column@width@\number\count@\endcsname%
  \advance\@tempdima\@tempdimb%
  \ifnum\@tempdimb=\z@%
    \advance\@tempcnta\@ne%
  \else%
    \PackageInfo{parcolumns}{Width of column \number\count@%
      \space set to \the\@tempdimb}
  \fi%
\repeat%
```

If at least one column has an unknown width:

如果至少有一列宽度未知：

```
\ifnum\@tempcnta>\z@%
```

`\@tempdimc` contains the distance between columns. We calculate the space left for the columns with an unknown width.

`@tempdimc` 包含列之间的间距。我们计算留给宽度未知的列的空间。

`\@tempdimb` will contain the sum of the space between all the columns …

`@tempdimb` 将包含所有列之间的距离的总和 …

```
    \@tempdimb\@tempdimc%
    \multiply\@tempdimb\pc@columncount%
    \advance\@tempdimb-\@tempdimc%
```

… plus the sum of the width of all columns with a known width:

… 加上所有已知宽度列的宽度之和：

```
    \advance\@tempdimb\@tempdima%
```

`\@tempdima` will contain the space for each column with an unknown width:

`@tempdima` 将包含每个未知宽度的列的空间：

```
    \@tempdima\linewidth%
    \advance\@tempdima-\@tempdimb%
    \divide\@tempdima\@tempcnta%
```

We set the widths of the colums with an unknown width:

设置未知宽度的列的宽度：

```
    \count@\z@%
    \loop\ifnum\count@<\pc@columncount%
      \advance\count@\@ne%
      \ifnum\csname pc@column@width@\number\count@\endcsname=\z@%
        \csname pc@column@width@\number\count@\endcsname\@tempdima%
        \PackageInfo{parcolumns}{Width of column \number\count@%
          \space calculated as \the\@tempdima}
      \fi%
    \repeat%
  \fi%
}
```

`\pc@setsinglecolwidth` Usage: `\pc@setsinglecolwidth {`⟨*column*⟩`} {`⟨*width*⟩`}`.

Sets the width of the column ⟨*column*⟩ to the width ⟨*width*⟩. Displays an error message if the column is not valid.

将第 ⟨*column*⟩ 列的宽度设置为 ⟨*width*⟩。如果列无效，则显示错误消息。

```
\def\pc@setsinglecolwidth#1#2{%
  \@ifundefined{pc@column@width@\number#1}{
    \PackageError{parcolumns}{`#1' is not a valid column number!}%
      {\@ehc}%
  }{%
    \csname pc@column@width@\number#1\endcsname=#2\relax%
  }%
}
```

`\pc@setcolumnwidth` Sets `\hsize` to the width of a column (stored in `\pc@column@width@`⟨*pc@columnctr*⟩) and enters horizonal mode.

将 `\hsize` 设置为列的宽度（存储在 `\pc@column@width@`⟨*pc@columnctr*⟩ 中），并进入水平模式。

```
\def\pc@setcolumnwidth{%
  \hsize\csname pc@column@width@\number\pc@columnctr\endcsname%
  \linewidth\hsize%
  \leavevmode%
}
```