

multicolrule — Decorative rules between columns*

Karl Hagen[†]

Released 2018/12/21

Abstract

The `multicolrule` package lets you customize the appearance of the vertical rule that appears between columns of multicolumn text. It is primarily intended to work with the `multicol` package, hence its name, but it also supports the `twocolumn` option and `\twocolumn` macro provided by the standard classes (and related classes such as the KOMA-Script equivalents), as well as the `bidi` package (and through it, all RTL scripts loaded with `polyglossia`).

Contents

1	Introduction	2
1.1	Bugs and Known Limitations	2
1.2	License	3
2	Package Options	3
2.1	Default Operation	3
2.2	Option ‘twocolumn’	3
2.3	Option ‘tikz’	3
3	The User Interface	4
3.1	Styles with the ‘line-style’ option	5
3.1.1	Notes on the Styles	6
3.1.2	Custom Patterns	6
3.2	Colors	7
3.3	Width	7
3.4	Repeated Rules	8
3.5	Extended Rules	8
4	Implementation	9
4.1	Preliminaries	9
4.2	Patching Output Routines	10
4.3	Creating the Rules	11
4.3.1	Tikz-only Routines	14
4.4	Color	16
4.5	Key-Values	16
4.6	User Interface	18

*This file describes version v1.1, last revised 2018/12/21.

[†]latex@polysyllabic.com

Change History**18****Index****19****1 Introduction**`line-style=dashed`

In \LaTeX , there are two lengths that control the formatting between columns of multicolumn text: `\columnsep` specifies the space between adjacent columns, and `\columnseprule` specifies the width of a solid vertical rule that is placed centered between the columns. The `multicol` package adds the ability to change the color of the rule, but in both vanilla \LaTeX and `multicol`, the rule itself is drawn directly inside the routines that output the column boxes, and is therefore difficult for users to alter.

Of course it's a legitimate question why anyone should *want* to change this rule, or indeed use one at all, as good typography tends to avoid using large vertical lines.¹ In my own case, I needed to modify the rule because of the requirements of a particular style I was imitating, and having done the hard work of creating the necessary infrastructure for one

line style, it was simple to extend the solution to a more general case. I hope someone else will find the options here useful.

The basic line styles that `multicolrule` makes available are illustrated throughout this guide. The default line-width used is 0.4pt (thin), and the default color is Maroon. You can also find examples of rules created with all available options in the file `mcrule-example.pdf`.

New for Version 1.1

Version 1.1 now supports drawing decorative rules if you have the `bidi` package loaded, which can occur automatically if you set a right-to-left language with `polyglossia`. It also provides a mechanism to extend or shrink rules by fixed amounts, as well as to have the rule fill the available space to the end of the text area (see section 3.5).

1.1 Bugs and Known Limitations`line-style=dots`

There are likely bugs that remain to be uncovered, as well as missing features and inefficient methods that should be improved upon. The development code is maintained on github (<https://github.com/polysyllabic/multicolrule>), and you can file feature requests or bug reports there. Alternatively, you can send an email to latex@polysyllabic.com. I welcome contributions for additional styles, especially to provide more options when running the package without `tikz`.

The line styles that work by repeating elements in a tiled pattern may have significant gaps at the end of columns, particularly for larger patterns. You can mitigate this prob-

lem slightly by tweaking the spaces above and below a pattern, but the basic problem is a side-effect of the way these patterns are implemented (with `\cleaders`), which means that only an integer number of copies can be produced. Lines drawn with `tikz` do not have this problem.

I have also noticed occasional instances, most noticeably when a `multicols` environment starts near the bottom of a page and the columns continue to the next one, where the rules are either somewhat shorter than they should be or shifted upward from where they belong. In the limited testing I have done, this appears to be a consequence of how `multicol` works, as the default rules show the same be-

¹See, for example, the remarks in the documentation for the `booktabs` package

havior. I may try to nail down this issue in future version, but as it's an edge case that disappears when you add page breaks or rewrite the text to alter how the columns are filled, it hasn't seemed worth taking the time to fix at this point.

This package works by patching the output routines of either `multicol` or the \LaTeX kernel, depending on the mode of operation. If `bidi` is loaded, it will also patch that. It will have no effect if you use a class or package that outputs column text via alternate mechanisms. This includes `parcolumns`, and probably other classes and packages designed

to typeset parallel-column text as well, although I have not done a survey to determine whether this is the case. If you would like support for one of these, please send me an email or file a feature request on github and I'll see what I can do.

`multicolrule` is written using `expl3` syntax, and so requires a less-than-ancient installation of \LaTeX . It uses the packages `l3keys2e`, `xparse`, `xpatch`, and `xcolor`, and depending on the mode of operation may also require `multicol` and `tikz`. If you have an up-to-date distribution, these requirements should cause no issues.

1.2 License

The `multicolrule` package is copyright 2018 by Karl Hagen. It may be distributed and/or modified under the conditions of the \LaTeX Project Public License, either version 1.3c of this license or (at your option) any later version.

The latest version of this license is in <http://www.latex-project.org/lppl.txt>.

This work has the LPPL maintenance status 'maintained.' The Current Maintainer of this work is Karl Hagen.

2 Package Options

2.1 Default Operation

If you load `multicolrule` with its default settings, it will enable `multicol` support, and that package will be loaded if it hasn't been already. Note that if you need to pass any parameters to `multicol`, such as `docolaction`, you should load `multicol` with the appropriate settings *before* you load `multicolrule`, as \LaTeX does not support reloading packages with different parameters.

Because `multicol` does not work well with the ordinary two-column mode, `multicolrule` is only designed to work with one or the other at a time. If you try to use the `twocolumn` option when `multicol` has already been loaded, you will receive a warning, and nothing is guaranteed. But the custom rules will at best only appear in the conventional two-column mode and not within a `multicols` environment.

2.2 Option 'twocolumn'

The `multicolrule` package recognizes the option `twocolumn`, either as a package option or as a global class option. If you pass this option to your document class, you do not need to pass it a second time to the package. It is only necessary to use the package option if you plan to have a predominantly one-column document and use `\twocolumn` to switch temporarily into two-column mode.

2.3 Option 'tikz'

You have access to a wider set of line styles if you also use the `tikz` package. Some line styles are only available if `tikz` is enabled, and others look better with it. The default behavior of `multicolrule` depends on the status of the `tikz` package at the time `multicolrule` is loaded. If `multicolrule` detects that `tikz` is already loaded, then `tikz` support will be enabled by default. Otherwise, you need the `tikz` to enable it. This option also ac-

`line-style=dotted,`
`width=ultra-thick`

`line-style=dash-dot`

cepts explicit boolean values, so you can pass `tikz=false` if you want to explicitly disable `tikz` support. If `tikz` support is not enabled (or if it is explicitly disabled), the line styles marked *tikz only* in section 3.1 will be unavailable and errors will result if you try to use them.

3 The User Interface

`line-style=circles,`
`width=2pt`

The `multicolrule` package has just a single user command:

```
\SetMCRule {⟨key-value⟩}
```

It takes one parameter containing a key-value list of all options you want to set. You can issue this command in the preamble or the document body. Changes to the rule settings are local to the current group. For example, if you call `\SetMCRule` inside a `multicols` environment, the rule settings will revert to their previous values once the environment ends. Also note that any changes made with

`\SetMCRule` when multiple columns are active will appear starting on the same page as your current location when you issue the command, and will extend the height of the full column box. It is not possible to have a rule change styles in the middle of a page unless you close out one `multicols` environment and begin another.

Table 1 summarizes the keys available in `\SetMCRule`. The functions of each is described in detail in the sections that follow.

Table 1: `\SetMCRule` keys

Key	Purpose
<code>color</code>	Set the color of the rule (see sec. 3.2)
<code>color-model</code>	Set the color model of the rule (see sec. 3.2)
<code>custom-line</code>	Set a custom <code>tikz</code> line for the rule (<i>tikz only</i> ; see sec. 3.1.2)
<code>custom-pattern</code>	Set a custom individual pattern for the rule (see sec. 3.1.2)
<code>custom-tile</code>	Set a custom tiling pattern for the rule (see sec. 3.1.2)
<code>double</code>	Draw two copies of the rule (see sec. 3.4)
<code>extend-bot</code>	Set an extra amount to extend the rule at the bottom of the column (see sec. 3.5)
<code>extend-fill</code>	Extend rule to the bottom of the text area (<i>multicol only</i> ; see sec. 3.5)
<code>extend-reserve</code>	Space to reserve at bottom of text area when using <code>extend-fill</code> (<i>multicol only</i> ; see sec. 3.5)
<code>extend-top</code>	Set an extra amount to extend the rule at the top of the column (see sec. 3.5)
<code>line-style</code>	Select the type of rule printed (default= <i>default</i> ; see sec. 3.1)
<code>single</code>	Draw a single copy of the rule (<i>default</i> ; see sec. 3.4)
<code>repeat</code>	Set the number of times to draw the rule (see sec. 3.4)
<code>repeat-distance</code>	Set the horizontal space between adjacent copies of repeated rules (see sec. 3.4)
<code>triple</code>	Draw three copies of the rule (see sec. 3.4)
<code>width</code>	Set the width of the rule (see sec. 3.3)

3.1 Styles with the ‘line-style’ option

```
line-style=
solid-circles,
width=4pt
```

You choose a style for the rule with the `line-style` key. If the predefined styles are insufficient, see section 3.1.2 for different ways to customize it. The width of most line styles depends on the setting of `\columnseprule`, the default L^AT_EX length that controls the width of the column rule (see section 3.3).

Table 2 summarizes the available line styles. Most of the basic patterns come in three versions, differing only in how closely the pattern is spaced: normal, dense, and loose. These settings parallel those found in `tikz`.

Table 2: Styles available for the line-style key

Style	Description
<code>circles</code>	A series of hollow circles (<i>tikz only</i>)
<code>dash-dot</code>	A dash followed by a square dot (<i>tikz only</i>)
<code>dash-dot-dot</code>	A dash followed by two square dots (<i>tikz only</i>)
<code>dashed</code>	A series of dashed lines
<code>default</code>	A solid rule drawn the same way as the default multicol rule. Does not support extended rules.
<code>dense-circles</code>	The same as <code>circles</code> but more closely spaced (<i>tikz only</i>)
<code>dense-dots</code>	The same as <code>dots</code> but more closely spaced
<code>dense-solid-circles</code>	The same as <code>solid-circles</code> but more closely spaced (<i>tikz only</i>)
<code>densely-dash-dot</code>	The same as <code>dash-dot</code> but more closely spaced (<i>tikz only</i>)
<code>densely-dash-dot-dot</code>	The same as <code>dash-dot-dot</code> but more closely spaced (<i>tikz only</i>)
<code>densely-dashed</code>	The same as <code>dashed</code> but more closely spaced
<code>densely-dotted</code>	The same as <code>dotted</code> but more closely spaced
<code>dots</code>	A series of dots drawn with the period (full-stop) of the current font
<code>dotted</code>	A series of square dots
<code>loose-dots</code>	The same as <code>dots</code> but spaced further apart
<code>loose-circles</code>	The same as <code>circles</code> but spaced further apart (<i>tikz only</i>)
<code>loose-solid-circles</code>	The same as <code>solid-circles</code> but spaced further apart (<i>tikz only</i>)
<code>loosely-dash-dot</code>	The same as <code>dash-dot</code> but spaced further apart (<i>tikz only</i>)
<code>loosely-dash-dot-dot</code>	The same as <code>dash-dot-dot</code> but spaced further apart (<i>tikz only</i>)
<code>loosely-dashed</code>	The same as <code>dashed</code> but spaced further apart
<code>loosely-dotted</code>	The same as <code>dotted</code> but spaced further apart
<code>solid</code>	A solid line, like <code>default</code> , but supports extending rules
<code>solid-circles</code>	A series of filled circles (<i>tikz only</i>)

3.1.1 Notes on the Styles

`line-style=solid`

The `default` and `solid` line styles are nearly the same, except that the `solid` line (as of version 1.1) supports the rule-extension commands described in section 3.5. This means that if you want a solid rule with altered top or bottom extensions, you must explicitly set the line style to `solid`. If you make no calls to `\SetMCRule` after loading `multicolrule`, the column divider will continue to behave exactly as it does with the ordinary `multicol` package.

You can alter the rule's width and color either through `\SetMCRule` with the `width` and `color` keys described in sections 3.3 and 3.2, respectively, or directly by changing the value of `\columnseprule` and renewing

the `\columnseprulecolor` macro. All line styles, including `default`, can be repeated as many times as you like (see section 3.4).

The `dots` style and its variants are rendered with a period (.) in the currently active font. This means that changing `\columnseprule` will not change the size of these dots, although, as with all rules, it will not appear at all if `\columnseprule` is set to 0pt. Custom tiles and patterns also do not scale with `\columnseprule`.

The `dotted` styles differ from `dots` in that the former are squares with side lengths equal to `\columnseprule`. This mirrors the behavior of the equivalently named dotted patterns in `tikz`.

3.1.2 Custom Patterns

`custom-tile= {<pattern>} {<space above>} {<space below>}`

`custom-tile=`
`{\SparkleBold}`
`{16pt}{16pt}`

There are three options to create custom rules with `multicolrule`. The first is the `custom-tile` key. This creates a rule consisting of vertically stacked boxes of arbitrary content—the tile—running the height of the column separator. The `custom-tile` key takes three parameters, which must all be enclosed brackets and may not be omitted. The first should contain the tokens you want to appear as the content of the tile. The second is a dimension specifying the leading vertical space to apply above each copy of the tile.



The third is a dimension specifying the trailing vertical space to insert below each copy of the tile.



The rule in this section uses the `\SparkleBold` symbol from `bbding`. Notice that when you use the `custom-tile` parameter, of any of the other custom key commands, you do *not* specify a separate `line-style`. If you try to provide both, the last style given in the list will be the one that is kept.



`custom-pattern= {<pattern>} {<shift down>} {<shift up>}`

`custom-pattern=`
`{\HandRight}`
`{0pt}{0pt}`

The second custom option is with the `custom-pattern` key. The syntax is identical to that for `custom-tile`, but the content you specify will appear once per page or column pair (if the columns occupy less than a full page). This content will be vertically cen-



tered if the second and third parameters are both 0pt. You can shift the content down by increasing the second parameter, and up by increasing the third. The rule in this section uses the `\HandRight` symbol from `bbding`.

```
custom-line= {\draw command}
```

```
custom-line={
\draw[line width=
\columnseprule] (TOP)
to [ornament=88]
(BOT);},
width=1pt
```

The third custom pattern involves setting your own tikz drawing function using the key `custom-line`. The rule in this section is drawn with an ornament from `pgforornaments`. Obviously, this feature requires tikz support. The value you provide to the `custom-line` key should consist of a tikz command, such as `\draw`, without the surrounding `tikzpicture` environment.

Before the drawing command is called, `multicolrule` will set up a `tikzpicture` with both the x- and y-coordinates scaled to points, and two nodes, named (TOP) and (BOT), which are set to the coordinates of the top and bottom of the rule. You can then specify

your own `\draw` function in whatever way you like. The rule separating these columns was drawn with a decorative element from the `pgforornaments` package.

This function will use the color set in `\columnseprulecolor` if you don't set it explicitly within the tikz command, but you must provide everything else necessary to draw the line correctly, including the line width. Note that this function should be considered experimental. It works for single-line commands such as the one shown in the example, but I haven't tested it with anything more elaborate.

3.2 Colors

```
line-style=solid,
width=2pt
color-model=cmy,
color={0.7,0.5,0.3}
```

You can set colors for the rule through the `color` and, optionally, the `color-model` keys. `multicolrule` loads the `xcolor` package to manage colors, and the `color` parameter accepts any name that `xcolor` recognizes, either natively or as the result of any names you have defined with `\definecolor` (see the `xcolor` documentation). Note that if you want to use color names that are defined through the one of `xcolor`'s package options, you must load `xcolor` before both `multicolrule` and `tikz` with the relevant options.

To specify a color by a numeric specification, you use the `color-model` parameter to specify any color model that `xcolor` recognizes (rgb, cmy, etc), and `color` to hold the

color-specification list. Because that list is itself comma-separated, you must enclose it in brackets.

The current color setting can always be found in `\columnseprulecolor`. If you are running in `twocolumn` mode without `multicol`, this command will be provided and colors will work the same way they do with `multicol`. Note that setting the `color` key causes `\columnseprulecolor` to be redefined within the current group only. If you directly redefine `\columnseprulecolor`, the color of the custom rule will reflect this setting. This way, the settings of any packages that might alter the rule color will be respected.

3.3 Width

```
line-style=
dash-dot-dot,
width=thick
```

You can set the width of the rule with the `width` key. Legal values are any explicit dimension or dimension expression, as well as with names that parallel those used by `tikz`, except that spaces in the key names are replaced with hyphens.

The current width of the rule is kept in `\columnseprule`, just as in vanilla \LaTeX , and if it is set separately, the custom rule's

width will reflect this change. Note that although some line styles do not depend directly on `\columnseprule` to calculate their actual width, the value of `\columnseprule` must be greater than 0pt for any rule to appear. This behavior is intentional and is in keeping with the way the default column rules work.

Table 3: Sizes of named line widths

Name	Width
<code>ultra-thin</code>	0.1pt
<code>very-thin</code>	0.2pt
<code>thin</code>	0.4pt
<code>semithick</code>	0.6pt
<code>thick</code>	0.8pt
<code>very-thick</code>	1.2pt
<code>ultra-thick</code>	1.6pt

3.4 Repeated Rules

```
line-style=
dash-dot-dot,
triple=2pt
```

You can draw multiple, adjacent copies of any rule by setting the number of times to draw the rule with the `repeat` key. The space between copies is controlled with the `repeat-distance` key. Initially, this distance is set to `\columnseprule`.

The keys `single`, `double`, and `triple` are shorthand methods to set the number of repeats and the `repeat-distance` at

the same time. If you use the key without a value `repeat-distance` is set to `\columnseprule`.

There are no checks made to ensure that repeated rules will fit in the available space between columns, so you should be careful using these commands, especially with thicker rules.

3.5 Extended Rules

```
line-style=dashed,
extend-top=-16pt,
extend-bot=-16pt
```

You can specify an additional amount by which the top or bottom of the rule projects beyond the column’s natural length with the keys `extend-top` and `extend-bot`, each of which can be set to a dimension expression. Extending the top of the rule with a positive dimension will push the columns down from any preceding material. A positive value for `extend-bot` does the same in the other direction when a column ends in the middle of a page, but the rule will extend into the the bottom margin if the column goes to the end of the page, and so you probably only want to use this in very limited situations where you need a special effect for one small `multicol` environment. Negative values for both keys may be more generally useful, as they have the effect of shrinking the rule. This behavior is illustrated with the rule for this section.

The `extend-fill` key is a boolean option that, when set to true, will extend the rule to occupy any space between the bottom

of the columns and the end of the text area. Providing the key with no value is equivalent to `extend-fill=true`. This option has no effect unless the `multicol` package is loaded.

If you want text below the `multicols` environment when using `extend-fill`, you can reserve space for it with `extend-reserve`, which takes a dimension expression specifying the vertical space to leave available after the rule. If the value is greater than zero, the height of the extended line will be reduced by the reserved amount plus the value of `\multicolsep`. In other words, you only have to specify the actual space you need for the text itself, not the space that `multicol` adds automatically below the columns. Note that if the amount you request for reserved space is less than the amount actually available at the end of the page, the rule will not extend below the columns and you probably will find this material spilling onto the next page anyway.

4 Implementation

```

1 <*package>
2 <@@=mcrule>

```

4.1 Preliminaries

```

3 \ProvidesExplPackage {multicolrule} {2018/12/21} {1.1}
4 {Decorative~vertical~rules~between~columns}

```

We always need these packages.

```

5 \RequirePackage{l3keys2e}
6 \RequirePackage{xpatch}
7 \RequirePackage{xcolor}
8 \RequirePackage{scrfile}

```

Define the messages we use.

```

9 \msg_new:nnn {multicolrule} {patch-success} {Patched~#1.}
10 \msg_new:nnn {multicolrule} {patch-failure} {Error~patching~#1.}
11 \msg_new:nnn {multicolrule} {tikz-required}
12 {The~'1'~setting~requires~tikz~to~work.~Either~load~tikz~before~you~load~
13  multicolrule~or~use~multicolrule's~'tikz'~package~option.}
14 \msg_new:nnn {multicolrule} {multicol-loaded} {You~are~using~the~'twocolumn'~
15  option~with~multicol~already~loaded.~You~will~likely~run~into~problems.}

```

```

\g__mcrule_twocolumn_bool
\g__mcrule_use_tikz_bool

```

Flags for package options

```

16 \bool_new:N \g__mcrule_twocolumn_bool
17 \bool_new:N \g__mcrule_use_tikz_bool

```

(End definition for `\g__mcrule_twocolumn_bool` and `\g__mcrule_use_tikz_bool`.)

```

\l__mcrule_repeat_int
\l__mcrule_repeat_distance_dim

```

Variables to support repeated copies of the rule.

```

18 \int_new:N \l__mcrule_repeat_int
19 \int_set:Nn \l__mcrule_repeat_int {1}
20 \dim_new:N \l__mcrule_repeat_distance_dim

```

(End definition for `\l__mcrule_repeat_int` and `\l__mcrule_repeat_distance_dim`.)

```

\l__mcrule_extend_top_dim
\l__mcrule_extend_bot_dim
\l__mcrule_extend_fill_bool
\l__mcrule_extend_reserve_dim

```

Variables to control the distance to extend the rule above and below the natural column height.

```

21 \dim_new:N \l__mcrule_extend_top_dim
22 \dim_new:N \l__mcrule_extend_bot_dim
23 \bool_new:N \l__mcrule_extend_fill_bool
24 \dim_new:N \l__mcrule_extend_reserve_dim

```

(End definition for `\l__mcrule_extend_top_dim` and others.)

```

\l__mcrule_color_name_tl
\l__mcrule_color_model_tl

```

Keep name and color model so we can set them separately while retaining the value of the other one.

```

25 \tl_new:N \l__mcrule_color_name_tl
26 \tl_new:N \l__mcrule_color_model_tl

```

(End definition for `\l__mcrule_color_name_tl` and `\l__mcrule_color_model_tl`.)

If `tikz` is already loaded, enable `tikz`-sensitive line styles unless the user explicitly disables them. If `tikz` is not already loaded, these functions are disabled unless they are explicitly loaded.

```

27 \@ifpackageloaded{tikz}
28 {
29   \bool_gset_true:N \g__mcrule_use_tikz_bool
30 }{}

```

Set up the keys for package options and process them.

```

31 \keys_define:nn {mcrule-opts}
32 {
33   twocolumn .bool_gset:N = \g__mcrule_twocolumn_bool,
34   tikz      .bool_gset:N = \g__mcrule_use_tikz_bool,
35   tikz      .default:n    = true,
36 }
37 \ProcessKeysOptions{mcrule-opts}

```

4.2 Patching Output Routines

`__mcrule_column_height:`
`__mcrule_column_depth:`

Get the height and depth of the box appropriate to the supported mode.

```

38 \cs_new:Npn \__mcrule_column_height: {}
39 \cs_new:Npn \__mcrule_column_depth: {}

```

Now that we know what mode we're going to run in, we patch the output routine(s) to substitute our custom rule for the vanilla one. Since multicol doesn't fully support twocolumn mode, we patch one or the other, but not both.

`__mcrule_patch_mcol_output:N`

```

40 \cs_new_protected:Npn \__mcrule_patch_mcol_output:N #1
41 {
42   \xpatchcmd{#1} {\vrule\@width\columnseprule} {\mcruledivider}
43   {\msg_info:nnn {multicolrule} {patch-success} {#1}}
44   {\msg_info:nnn {multicolrule} {patch-failure} {#1}}
45 }

```

`__mcrule_patch_twocol_output:N`

```

46 \cs_new_protected:Npn \__mcrule_patch_twocol_output:N #1
47 {
48   \xpatchcmd{#1} {\normalcolor\vrule\@width\columnseprule}
49   {\columnseprulecolor\mcruledivider}
50   {\msg_info:nnn {multicolrule} {patch-success} {#1}}
51   {\msg_info:nnn {multicolrule} {patch-failure} {#1}}
52 }
53 \bool_if:NTF \g__mcrule_twocolumn_bool
54 {
55   \@ifpackageloaded{multicol}
56   {\msg_warning:nn {multicolrule} {multicol-loaded}}{}

```

Provide the column-color macro from multicol.

```

57 \cs_gset:Npn \columnseprulecolor {\normalcolor}
58 \cs_gset:Npn \__mcrule_column_height: {\box_ht:N \@outputbox}
59 \cs_gset:Npn \__mcrule_column_depth: {\box_dp:N \@outputbox}
60 \__mcrule_patch_twocol_output:N \@outputdblcol

```

Now patch the relevant code in `\@outputdblcol`, replacing the hard-coded rule with a macro that we can overwrite.

```

61 \__mcrule_patch_twocol_output:N \@outputdblcol

```

bidi has two output routines to patch, and it insists on being loaded after xcolor, tikz, *and* multicol, so it must always be loaded after us. We use \AfterPackage from scrfile to insert the patch if bidi is loaded later on.

```

62 \AfterPackage!{bidi}
63 {
64   \__mcrule_patch_twocol_output:N \RTL@outputdblcol
65   \__mcrule_patch_twocol_output:N \LTR@outputdblcol
66 }
67 }
```

Now patch for multicol.

```

68 {
69   \RequirePackage{multicol}
70   \__mcrule_patch_mcol_output:N \LR@column@boxes
71   \__mcrule_patch_mcol_output:N \RL@column@boxes
```

Although taking the height of \mult@rightbox is a reliable way to get the column height, the same isn't true for the depth, so we use \dimen\tw@, which multicol uses to hold the maximum depth of all the columns, instead.

```

72 \cs_gset:Npn \__mcrule_column_height: {\box_ht:N \mult@rightbox}
73 \cs_gset:Npn \__mcrule_column_depth: {\dimen\tw@}
```

We need to reissue \LRmulticolcolumns to update the actual code in \mc@align@columns.

```

74 \LRmulticolcolumns
```

The bidi package supplies its own versions of most core multicol functions, including the output boxes. Much of this is unnecessary, as current versions of multicol support printing the columns in right-to-left order, and the effect is to leave the original multicol definitions loaded but unused. As a result, after these changes, the multicol commands \LRmulticolcolumns and \RLmulticolcolumns have no visible effect. But as bidi also reworks the footnotes extensively, it's easier just to patch the equivalent output routines rather than rewrite it properly.

```

75 \AfterPackage!{bidi}
76 {
77   \cs_gset_eq:NN \LTR@column@boxes \LR@column@boxes
78   \cs_gset_eq:NN \RTL@column@boxes \RL@column@boxes
```

While we're at it, we also redefine \LRmulticolcolumns and \RLmulticolcolumns so they work the way people expect them to.

```

79 \cs_gset_eq:NN \LRmulticolcolumns \LTRmulticolcolumns
80 \cs_gset_eq:NN \RLmulticolcolumns \RTLmulticolcolumns
81 }
82 }
```

4.3 Creating the Rules

Utility functions for different rule types

`\mcruledivider` This is the function directly called by the patched output routines. It's given a $\LaTeX 2$ name so the user can redefine it if necessary. Its main function is to call the internal function `__mcrule_divider:`, which contains the actual rule-typesetting instructions, the number of times specified in `\l__mcrule_repeat_int`. We only call `__mcrule_divider:` if `\columnseprule > 0`, so that all line styles can be turned off by setting it to 0, just as is the case with the vanilla rules.

```

83 \cs_new:Npn \mcruledivider
84 {
85   \bool_lazy_and:nnT
86   {\dim_compare_p:nNn {\columnseprule} > {0pt}}
87   {\int_compare_p:nNn {\l__mcrule_repeat_int} > {0}}
88   {
89     \__mcrule_divider:
90     \prg_replicate:nn {\l__mcrule_repeat_int - 1}
91     {
92       \hspace{\l__mcrule_repeat_distance_dim}
93       \__mcrule_divider:
94     }
95   }
96 }

```

(End definition for \mcruledivider. This function is documented on page ??.)

```

\__mcrule_column_total_height:
\__mcrule_column_total_depth:

```

Get column height and depth with any explicit alterations.

```

97 \cs_new:Npn \__mcrule_column_total_height:
98 {
99   \dim_eval:n {\__mcrule_column_height: + \__mcrule_column_depth: +
100     \__mcrule_extend_column_top: + \__mcrule_extend_column_bottom:}
101 }
102 \cs_new:Npn \__mcrule_column_total_depth:
103 {
104   \dim_eval:n {\__mcrule_column_depth: + \__mcrule_extend_column_bottom:}
105 }

```

```

\__mcrule_extend_column_top:

```

Currently, the extend amount for the top is just the \l__@@_extend_top_dim distance. In the future we may allow more complex criteria, such as by odd or even page, or on a particular page. Although these might theoretically be useful, I'm not going to implement them until someone comes along with a use-case for it.

```

106 \cs_new:Npn \__mcrule_extend_column_top:
107 {
108   \l__mcrule_extend_top_dim
109 }

```

`_mcrule_extend_column_bottom:`

The `extend-fill` option, which is only applicable with `multicol`, extends the rule from the bottom of the column to the end of the text area, minus whatever reserved space the user specifies. If there's less space available than requested, we give everything we can.

```

110 \cs_new:Npn \\_mcrule_extend_column_bottom:
111 {
112   \bool_lazy_and:nnTF
113   {\bool_if_p:n {\l__mcrule_extend_fill_bool}}
114   {\bool_not_p:n {\g__mcrule_twocolumn_bool}}
115   {
116     \dim_compare:nNnTF
117     {\@colroom - \\_mcrule_column_height: - \\_mcrule_extend_reserve:} > {0pt}
118     {\@colroom - \\_mcrule_column_height: - \\_mcrule_extend_reserve:}
119     {0pt}
120   }
121   {\l__mcrule_extend_bot_dim}
122 }
```

`_mcrule_extend_reserve:`

The reserved space is the amount of user-provided space we want, but we also have to account for the space added with `\multicolsep`.

```

123 \cs_new:Npn \\_mcrule_extend_reserve:
124 {
125   \dim_compare:nNnTF {\l__mcrule_extend_reserve_dim} > {0pt}
126   {\dim_eval:n {\l__mcrule_extend_reserve_dim + \multicolsep}}
127   {0pt}
128 }
```

`_mcrule_divider:`

This is the internal routine that contains the instructions to draw one copy of rule between columns. The default is identical to the original definition used by `multicol`. It will be reset each time the user calls `\MCSetRule`.

```

129 \cs_new:Npn \\_mcrule_divider: {\vrule\@width\columnseprule}
```

`_mcrule_pattern:nnn`

```

\\_mcrule_pattern:nnn {\langle pattern \rangle} {\langle space above \rangle} {\langle space below \rangle}
```

Typesets a single copy of a pattern, vertically centered, in a vertical box that is the height of the current column. The pattern must be something that can go in a horizontal box. The spaces above and below must be fixed dimensions.

```

130 \cs_new_nopar:Npn \\_mcrule_pattern:nnn #1#2#3
131 {
132   \box_move_down:nn {\_mcrule_column_total_depth:}
133   {
134     \vbox_to_ht:nn {\_mcrule_column_total_height:}
135     {
136       \tex_vfill:D
137       \tex_kern:D #2 \hbox:n{#1} \tex_kern:D #3
138       \tex_vfill:D
139     }
140   }
141 }
```

```

\__mcrule_tile_pattern:nnn \__mcrule_tile_pattern:nnn {\langle pattern \rangle} {\langle space above \rangle} {\langle space below \rangle}

```

Typesets multiple copies of pattern, tiled so as to occupy a vertical box that is the height of the current column. The pattern must be something that can go in a horizontal box. The spaces above and below must be fixed dimensions.

```

142 \cs_new_nopar:Npn \__mcrule_tile_pattern:nnn #1#2#3
143 {
144   \box_move_down:nn {\__mcrule_column_total_depth:}
145   {
146     \vbox_to_ht:nn {\__mcrule_column_total_height:}
147     {
148       \tex_cleaders:D \vbox:n
149       {
150         \tex_kern:D #2 \hbox:n{#1} \tex_kern:D #3
151       }
152       \tex_vfill:D
153     }
154   }
155 }

```

```

\__mcrule_line_pattern:nnnn \__mcrule_line_pattern:nnnn {\langle tikz-name \rangle} {\langle height \rangle} {\langle space above \rangle}
{\langle space below \rangle}

```

This function can draw a line pattern using either a tikz name or directly (as a tiled pattern). The latter case is currently limited to line patterns that can be described in terms of a solid line of length $\langle height \rangle$ separated by spaces above and/or below the line.

```

156 \cs_new:Npn \__mcrule_line_pattern:nnnn #1#2#3#4
157 {
158   \bool_if:NTF \g__mcrule_use_tikz_bool
159   {
160     \__mcrule_pattern_line:n {#1}
161   }
162   {
163     \__mcrule_tile_pattern:nnn {\rule{\columnseprule}{#2}}{#3}{#4}
164   }
165 }

```

```

\__mcrule_solid_line:

```

Unlike the default solid line, which is created with a simple `\vrule`, this version allows us to extend the line beyond the natural space of the column.

```

166 \cs_new:Npn \__mcrule_solid_line:
167 {
168   \rule[-\__mcrule_column_total_depth:]{\columnseprule}{\__mcrule_column_total_height:}
169 }

```

4.3.1 Tikz-only Routines

If we're supporting tikz, make sure it's loaded and redefine the relevant functions. We turn off `expl3` syntax to load the package because tikz relies on 2e catcodes, especially for spaces.

```

170 \bool_if:NTF \g__mcrule_use_tikz_bool
171 {
172   \ExplSyntaxOff
173   \RequirePackage{tikz}

```

```
174 \ExplSyntaxOn
```

```
\__mcrule_tikz_picture:n \__mcrule_tikz_picture:n {<draw function>}
```

Set up the `tikzpicture` environment and declare two nodes, named (TOP) and (BOT). This way we can pass a `\draw` routine directly, without worrying about the line's coordinates.

```
175 \cs_set:Npn \__mcrule_tikz_picture:n #1
176 {
177   \begin{tikzpicture}[x=1pt,y=1pt,inner~sep=0pt,outer~sep=0pt,
178     baseline={([yshift=\__mcrule_column_total_depth:]current~bounding~box.south)}]
179   \node (TOP) at (0,\__mcrule_column_total_height:) {};
180   \node (BOT) at (0,0) {};
181   #1
182   \end{tikzpicture}
183 }
```

```
\__mcrule_pattern_line:n \__mcrule_pattern_line:n {<tikz pattern>}
```

For the `tikz` versions of the predefined lines, we just draw a line the length of the column box. `<tikz pattern>` should contain the name of a line style that `tikz` recognizes.

```
184 \cs_set:Npn \__mcrule_pattern_line:n #1
185 {
186   \begin{tikzpicture}[x=1pt,y=1pt,inner~sep=0pt,outer~sep=0pt,
187     baseline={([yshift=\__mcrule_column_total_depth:]current~bounding~box.south)}]
188   \draw[line~width=\columnseprule,#1] (0,\__mcrule_column_total_height:) -- (0,0);
189   \end{tikzpicture}
190 }
```

```
\__mcrule_circle:
```

Draw a hollow circle with a diameter equal to `\columnseprule`. This will be used as a tile pattern.

```
191 \cs_set:Npn \__mcrule_circle:
192 {
193   \begin{tikzpicture}[x=1pt,y=1pt,inner~sep=0pt,outer~sep=0pt]
194   \draw (0,0) circle[radius=.5\columnseprule];
195   \end{tikzpicture}
196 }
```

```
\__mcrule_solid_circle:
```

Draw a filled circle with a diameter equal to `\columnseprule`. This will be used as a tile pattern.

```
197 \cs_set:Npn \__mcrule_solid_circle:
198 {
199   \begin{tikzpicture}[x=1pt,y=1pt,inner~sep=0pt,outer~sep=0pt]
200   \fill (0,0) circle[radius=.5\columnseprule];
201   \end{tikzpicture}
202 }
203 }
```

In case `tikz` functions are not active, we provide stubs that issue error messages.

```
204 {
205   \cs_set:Npn \__mcrule_tikz_picture:n #1
```



```

206     {\msg_error:nnn {multicolrule} {tikz-required} {#1}}
207 \cs_new:Npn \__mcrule_pattern_line:n #1
208     {\msg_error:nnn {multicolrule} {tikz-required} {#1}}
209 \cs_new:Npn \__mcrule_circle:
210     {\msg_error:nnn {multicolrule} {tikz-required} {circles}}
211 \cs_new:Npn \__mcrule_solid_circle:
212     {\msg_error:nnn {multicolrule} {tikz-required} {solid-circles}}
213 }

```

4.4 Color

`__mcrule_set_rule_color:`

Reset color definition in `\columnseprulecolor` by name or by model and color specification.

```

214 \cs_new_protected:Npn \__mcrule_set_rule_color:
215 {
216   \tl_if_empty:NT \l__mcrule_color_name_tl
217   {
218     \tl_set:Nn \l__mcrule_color_name_tl {black}
219   }
220   \tl_if_empty:NTF \l__mcrule_color_model_tl
221   {
222     \cs_set:Npn \columnseprulecolor {\color{\l__mcrule_color_name_tl}}
223   }
224   {
225     \cs_set:Npn \columnseprulecolor
226       {\color[\l__mcrule_color_model_tl]{\l__mcrule_color_name_tl}}
227   }
228 }

```

4.5 Key-Values

Set up all the key definitions. For the line styles, this involves resetting `__mcrule_divider:` to an appropriate value.

```

229 \keys_define:nn {mcrule}
230 {
231   extend-top           .dim_set:N = \l__mcrule_extend_top_dim,
232   extend-bot          .dim_set:N = \l__mcrule_extend_bot_dim,
233   extend-fill         .bool_set:N = \l__mcrule_extend_fill_bool,
234   extend-fill         .default:n = true,
235   extend-reserve      .dim_set:N = \l__mcrule_extend_reserve_dim,
236   line-style          .choice:,
237   line-style / default .code:n = \cs_set:Npn \__mcrule_divider:
238     {\vrule\@width\columnseprule},
239   line-style / solid   .code:n = \cs_set:Npn \__mcrule_divider:
240     {\__mcrule_solid_line:},
241   line-style / dots    .code:n = \cs_set:Npn \__mcrule_divider:
242     {\__mcrule_tile_pattern:nnn {.}{1pt}{1pt}},
243   line-style / dense-dots .code:n = \cs_set:Npn \__mcrule_divider:
244     {\__mcrule_tile_pattern:nnn {.}{1pt}{0pt}},
245   line-style / loose-dots .code:n = \cs_set:Npn \__mcrule_divider:
246     {\__mcrule_tile_pattern:nnn {.}{2pt}{2pt}},

```

```

247 line-style / circles .code:n = \cs_set:Npn \__mcrule_divider:
248   {\__mcrule_tile_pattern:nnn {\__mcrule_circle:}{1pt}{1pt}},
249 line-style / dense-circles .code:n = \cs_set:Npn \__mcrule_divider:
250   {\__mcrule_tile_pattern:nnn {\__mcrule_circle:}{1pt}{0pt}},
251 line-style / loose-circles .code:n = \cs_set:Npn \__mcrule_divider:
252   {\__mcrule_tile_pattern:nnn {\__mcrule_circle:}{2pt}{2pt}},
253 line-style / solid-circles .code:n = \cs_set:Npn \__mcrule_divider:
254   {\__mcrule_tile_pattern:nnn {\__mcrule_solid_circle:}{1pt}{1pt}},
255 line-style / dense-solid-circles .code:n = \cs_set:Npn \__mcrule_divider:
256   {\__mcrule_tile_pattern:nnn {\__mcrule_solid_circle:}{1pt}{0pt}},
257 line-style / loose-solid-circles .code:n = \cs_set:Npn \__mcrule_divider:
258   {\__mcrule_tile_pattern:nnn {\__mcrule_solid_circle:}{2pt}{2pt}},
259 line-style / dotted .code:n = \cs_set:Npn \__mcrule_divider:
260   {\__mcrule_line_pattern:nnnn {dotted}{\columnseprule}{1pt}{1pt}},
261 line-style / densely-dotted .code:n = \cs_set:Npn \__mcrule_divider:
262   {\__mcrule_line_pattern:nnnn {densely-dotted}{\columnseprule}{1pt}{0pt}},
263 line-style / loosely-dotted .code:n = \cs_set:Npn \__mcrule_divider:
264   {\__mcrule_line_pattern:nnnn {loosely-dotted}{\columnseprule}{2pt}{2pt}},
265 line-style / dashed .code:n = \cs_set:Npn \__mcrule_divider:
266   {\__mcrule_line_pattern:nnnn {dashed}{3pt}{1.5pt}{1.5pt}},
267 line-style / densely-dashed .code:n = \cs_set:Npn \__mcrule_divider:
268   {\__mcrule_line_pattern:nnnn {densely-dashed}{3pt}{1pt}{1pt}},
269 line-style / loosely-dashed .code:n = \cs_set:Npn \__mcrule_divider:
270   {\__mcrule_line_pattern:nnnn {loosely-dashed}{3pt}{3pt}{3pt}},
271 line-style / dash-dot .code:n = \cs_set:Npn \__mcrule_divider:
272   {\__mcrule_pattern_line:n{dash-dot}},
273 line-style / densely-dash-dot .code:n = \cs_set:Npn \__mcrule_divider:
274   {\__mcrule_pattern_line:n{densely-dash-dot}},
275 line-style / loosely-dash-dot .code:n = \cs_set:Npn \__mcrule_divider:
276   {\__mcrule_pattern_line:n{loosely-dash-dot}},
277 line-style / dash-dot-dot .code:n = \cs_set:Npn \__mcrule_divider:
278   {\__mcrule_pattern_line:n{dash-dot-dot}},
279 line-style / densely-dash-dot-dot .code:n = \cs_set:Npn \__mcrule_divider:
280   {\__mcrule_pattern_line:n{densely-dash-dot-dot}},
281 line-style / loosely-dash-dot-dot .code:n = \cs_set:Npn \__mcrule_divider:
282   {\__mcrule_pattern_line:n{loosely-dash-dot-dot}},
283 color .code:n = {
284   \tl_set:Nn \l__mcrule_color_name_tl {#1}
285   \__mcrule_set_rule_color:
286 },
287 color-model .code:n = {
288   \tl_set:Nn \l__mcrule_color_model_tl {#1}
289   \__mcrule_set_rule_color:
290 },
291 custom-line .code:n = \cs_set:Npn \__mcrule_divider:
292   {\__mcrule_tikz_picture:n {#1}},
293 custom-pattern .code:n = \cs_set:Npn \__mcrule_divider:
294   {\__mcrule_pattern:nnn #1},
295 custom-tile .code:n = \cs_set:Npn \__mcrule_divider:
296   {\__mcrule_tile_pattern:nnn #1},
297 width .choice:,
298 width / ultra-thin .code:n = \dim_set:Nn \columnseprule {0.1pt},
299 width / very-thin .code:n = \dim_set:Nn \columnseprule {0.2pt},
300 width / thin .code:n = \dim_set:Nn \columnseprule {0.4pt},

```

```
301 width / semithick .code:n = \dim_set:Nn \columnseprule {0.6pt},
302 width / thick .code:n = \dim_set:Nn \columnseprule {0.8pt},
303 width / very-thick .code:n = \dim_set:Nn \columnseprule {1.2pt},
304 width / ultra-thick .code:n = \dim_set:Nn \columnseprule {1.6pt},
305 width / unknown .code:n = {\dim_set:Nn \columnseprule {#1}},
306 repeat .int_set:N = \l__mcrule_repeat_int,
307 repeat-distance .dim_set:N = \l__mcrule_repeat_distance_dim,
308 single .meta:n = {
309   repeat = 1,
310   repeat-distance = #1
311 },
312 single .default:n = \columnseprule,
313 double .meta:n = {
314   repeat = 2,
315   repeat-distance = #1
316 },
317 double .default:n = \columnseprule,
318 triple .meta:n = {
319   repeat = 3,
320   repeat-distance = #1
321 },
322 triple .default:n = \columnseprule,
323 }
```

4.6 User Interface

With only one command, this section is short. All we do is set whatever keys the user passes. All the real work is done in the definitions above.

```
\SetMCRule \SetMCRule {<key-value list>}
324 \NewDocumentCommand{\SetMCRule}{m}
325 {
326   \keys_set:nn {mcrule} {#1}
327 }
```

(End definition for \SetMCRule. This function is documented on page ??.)

Change History

v1.0		v1.1
General: Initial public release	1	General: Work with bidi and allow extending rules 1

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

- A**
- `\AfterPackage` 11
- C**
- `\cleaders` 2, 148
- `\color` 222, 226
- `\columnsep` 2
- `\columnseprule` 2,
5–8, 11, 42, 48, 86, 129, 163, 168, 188,
194, 200, 238, 260, 262, 264, 298, 299,
300, 301, 302, 303, 304, 305, 312, 317, 322
- `\columnseprulecolor` 6, 7, 16, 49, 57, 222, 225
- D**
- `\definecolor` 7
- `\draw` 7, 15, 188, 194
- H**
- `\HandRight` 6
- L**
- `\LRmulticolcolumns` 11, 11, 11, 74, 79
- `\LTRmulticolcolumns` 79
- M**
- mcrule internal commands:
- `__mcrule_circle:`
..... 15, 191, 209, 248, 250, 252
- `l__mcrule_color_model_tl`
..... 25, 220, 226, 288
- `l__mcrule_color_name_tl`
..... 25, 216, 218, 222, 226, 284
- `__mcrule_column_depth:`
..... 10, 39, 59, 73, 99, 104
- `__mcrule_column_height:`
..... 10, 38, 58, 72, 99, 117, 118
- `__mcrule_column_total_depth:`
..... 12, 102, 132, 144, 168, 178, 187
- `__mcrule_column_total_height:`
..... 12, 97, 134, 146, 168, 179, 188
- `__mcrule_divider:`
..... 11, 13, 16, 89, 93, 129, 237,
239, 241, 243, 245, 247, 249, 251, 253,
255, 257, 259, 261, 263, 265, 267, 269,
271, 273, 275, 277, 279, 281, 291, 293, 295
- `l__mcrule_extend_bot_dim`
..... 21, 121, 232
- `__mcrule_extend_column_`
bottom: 13, 100, 104, 110
- `__mcrule_extend_column_top:` .
..... 12, 100, 106
- `l__mcrule_extend_fill_bool` ..
..... 21, 113, 233
- `__mcrule_extend_reserve:`
..... 13, 117, 118, 123
- `l__mcrule_extend_reserve_dim`
..... 21, 125, 126, 235
- `l__mcrule_extend_top_dim`
..... 21, 108, 231
- `__mcrule_line_pattern:nnnn` ..
..... 14, 156, 260, 262, 264, 266, 268, 270
- `__mcrule_patch_mcol_output:N`
..... 10, 40, 70, 71
- `__mcrule_patch_twocol_output:N`
..... 10, 46, 60, 61, 64, 65
- `__mcrule_pattern:nnn` .. 13, 130, 294
- `__mcrule_pattern_line:n` ... 15,
160, 184, 207, 272, 274, 276, 278, 280, 282
- `l__mcrule_repeat_distance_dim`
..... 18, 92, 307
- `l__mcrule_repeat_int` 11, 18, 87, 90, 306
- `__mcrule_set_rule_color:`
..... 16, 214, 285, 289
- `__mcrule_solid_circle:`
..... 15, 197, 211, 254, 256, 258
- `__mcrule_solid_line:` .. 14, 166, 240
- `__mcrule_tikz_picture:n`
..... 15, 175, 205, 292
- `__mcrule_tile_pattern:nnn` ...
..... 14, 142, 163, 242,
244, 246, 248, 250, 252, 254, 256, 258, 296
- `g__mcrule_twocolumn_bool`
..... 16, 33, 53, 114
- `g__mcrule_use_tikz_bool`
..... 16, 29, 34, 158, 170
- `\mcruledivider` 42, 49, 83
- `\MCSetRule` 13
- `\multicolsep` 8, 13, 126
- O**
- options:
- `tikz` 3
- `twocolumn` 3
- R**
- `\RLmulticolcolumns` 11, 11, 80
- `\RTLmulticolcolumns` 80

S	
<code>\SetMCRule</code>	4, 6, 324
<code>\SparkleBold</code>	6
T	
T _E X and L ^A T _E X 2 _ε commands:	
<code>\@colroom</code>	117, 118
<code>\@outputbox</code>	58, 59
<code>\@outputdblcol</code>	60, 61
<code>\LR@column@boxes</code>	70, 77
<code>\LTR@column@boxes</code>	77
<code>\LTR@outputdblcol</code>	65
<code>\mc@align@columns</code>	11
<code>\mult@rightbox</code>	11, 72
<code>\RL@column@boxes</code>	71, 78
<code>\RTL@column@boxes</code>	78
<code>\RTL@outputdblcol</code>	64
<code>\vrule</code>	14
<code>tikz (option)</code>	3
<code>twocolumn (option)</code>	3
<code>\twocolumn</code>	1, 3