

# org-special-block-extras

Musa Al-hassy

April 17, 2020

## Contents

1	Example Use	1
2	Core Utility	1
3	Colours	4
4	Parallel	5

## 1 Example Use

User type something along the lines of the following.

```
#+begin_red org
/This/
    *text*
        _is_
            red!
#+end_red
```

Which generates *red* text when exported to HTML and L<sup>A</sup>T<sub>E</sub>X, **while supporting Org markup**.

This article may be read as a PDF or as HTML or as pure Org!

The remaining sections are implementation matter.

## 2 Core Utility

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; Core utility
```

```
(defun org-special-block-extras--advice (backend blk contents _)
  "Invoke the appropriate custom block handler, if any."
```

A given custom block BLK has a TYPE extracted from it, then we send the block CONTENTS along with the current export BACKEND to the formatting function ORG-SPECIAL-BLOCK-EXTRAS/TYPE if it is defined, otherwise, we leave the CONTENTS of the block as is."

```
(let* ((type (nth 1 (nth 1 blk)))
      (handler (intern (format "org-special-block-extras--%s" type))))
  (ignore-errors (apply handler backend contents nil)))
```

<code>#+begin_blue</code>	This text is blue!		This text is black!
This text is blue!			This text is blue!
<code>#+end_blue</code>	This text is brown!		This text is brown!
<code>#+begin_brown</code>	This text is cyan!		This text is cyan!
This text is brown!	This text is darkgray!		This text is darkgray!
<code>#+end_brown</code>	This text is gray!		This text is gray!
<code>#+begin_cyan</code>	This text is green!		This text is green!
This text is cyan!	This text is lime!		This text is lime!
<code>#+end_cyan</code>	This text is magenta!		This text is magenta!
<code>#+begin_darkgray</code>	This text is olive!		This text is olive!
This text is <u>darkgray</u> !	This text is orange!		This text is orange!
<code>#+end_darkgray</code>	This text is pink!		This text is pink!
<code>#+begin_gray</code>	This text is purple!		This text is purple!
This text is <u>gray</u> !	This text is red!		This text is red!
<code>#+end_gray</code>	This text is teal!		This text is teal!
<code>#+begin_green</code>			This text is violet!
This text is green!			
<code>#+end_green</code>			
<code>#+begin_lightgray</code>			
This text is <u>lightgray</u> !			
<code>#+end_lightgray</code>			
<code>#+begin_lime</code>			

Figure 1: Write Org-markup once, generate for many backends ^\_^

```
(advice-add #'org-html-special-block :before-until
  (-partial #'org-special-block-extras--advice 'html))

(advice-add #'org-latex-special-block :before-until
  (-partial #'org-special-block-extras--advice 'latex))
```

Example:

```
#+begin_3parallel org
one

#+latex: \columnbreak
two
```

```
#+latex: \columnbreak
three
#+end_3parallel
```

Yields:

one | two | three

The screenshot shows the Emacs editor interface. The title bar reads "Emacs-x86\_64-10\_10 — (94 x 24)". The main window displays the source code for the example, with line numbers 243 to 255. The code is as follows:

```
243 Example:
244 <=> org
245 #+begin_3parallel org
246 one
247
248 #+latex: \columnbreak
249 two
250
251 #+latex: \columnbreak
252 three
253 #+end_3parallel|
254 <=
255
```

Below the main window, a buffer window titled "org-special-block-extras.org" is visible. It shows the rendered output of the example, which is identical to the one shown in the top part of the figure.

Figure 2: Displaying thoughts side-by-side ^\_^

### 3 Colours

<code>#+begin_blue</code>	This text is blue!
<code>This text is blue!</code>	
<code>#+end_blue</code>	This text is brown!
<code>#+begin_brown</code>	This text is cyan!
<code>This text is brown!</code>	This text is darkgray!
<code>#+end_brown</code>	
<code>#+begin_cyan</code>	This text is gray!
<code>This text is cyan!</code>	This text is green!
<code>#+end_cyan</code>	
<code>#+begin_darkgray</code>	This text is lightgray!
<code>This text is darkgray!</code>	This text is lime!
<code>#+end_darkgray</code>	
<code>#+begin_gray</code>	This text is magenta!
<code>This text is gray!</code>	This text is olive!
<code>#+end_gray</code>	
<code>#+begin_green</code>	This text is orange!
<code>This text is green!</code>	This text is pink!
<code>#+end_green</code>	
<code>#+begin_lightgray</code>	This text is purple!
<code>This text is lightgray!</code>	This text is red!
<code>#+end_lightgray</code>	
<code>#+begin_lime</code>	This text is teal!

This text is black!  
This text is blue!  
This text is brown!  
This text is cyan!  
This text is darkgray!  
This text is gray!  
This text is green!  
This text is lightgray!  
This text is lime!  
This text is magenta!  
This text is olive!  
This text is orange!  
This text is pink!  
This text is purple!  
This text is red!  
This text is teal!  
This text is violet!  
  
This text is yellow!

This text is black!  
 This text is blue!  
 This text is brown!  
 This text is cyan!  
 This text is darkgray!  
 This text is gray!  
 This text is green!  
 This text is lightgray!  
 This text is lime!  
 This text is magenta!  
 This text is olive!  
 This text is orange!  
 This text is pink!  
 This text is purple!  
 This text is red!  
 This text is teal!  
 This text is violet!  
  
 This text is yellow!

```

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;; Load support for 19 colour custom blocks

(defvar org-special-block-extras--colors
  '(black blue brown cyan darkgray gray green lightgray lime
    magenta olive orange pink purple red teal violet white
    yellow)
  "Colours that should be available on all systems.")

(loop for colour in org-special-block-extras--colors
  do (eval (read (format
    "(defun org-special-block-extras--%s (backend contents)
      (format (pcase backend
        ('latex \"\\\\\\begingroup\\\\\\color{%s}%s\\\\\\endgroup\")
        ('html  \"<div style=\\\\"color:%s;\\\\">%s</div>\\")
        (t      \"org-special-block-extras: Unsupported backend\\")
        contents)))
      colour colour colour))))

```

## 4 Parallel

Example:

```

#+begin_3parallel org
one

#+latex: \columnbreak
two

#+latex: \columnbreak
three
#+end_3parallel

```

Yields:

one | two | three

Example:

```
#+begin_3parallel org
one

#+latex: \columnbreak
two
```

```
#+latex: \columnbreak
three
#+end_3parallel
```

Yields:

one | two | three

The screenshot shows the Emacs editor interface. The title bar reads "Emacs-x86\_64-10\_10 — (94 x 24)". The main window displays the source code for the example, with line numbers 243 to 255. The code is as follows:

```
243 Example:
244 <=> org
245 #+begin_3parallel org
246 one
247
248 #+latex: \columnbreak
249 two
250
251 #+latex: \columnbreak
252 three
253 #+end_3parallel|
254 <=
255
```

Below the main window, a buffer window titled "org-special-block-extras.org" is visible, showing the rendered output of the code. It includes the "Example:" section and the "Yields:" section with the text "one | two | three". The buffer window also shows "Git:master Mod", "10 lines", "253:15", and "46%".

Figure 3: Displaying thoughts side-by-side ^\_^

I initially used the names `parallel<n>` but names ending with a number did not inherit highlighting, so I shifted the number to being a prefix instead.

- For L<sup>A</sup>T<sub>E</sub>X, new lines are used to suggest opportunities for column breaks and are needed even if explicit columnbreaks are declared.

```

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;; Parallel blocks: parallel<n>[NB] for n:2..5, optionally with 'N'o 'b'ar
;; in-between the columns.
;;
;; Common case is to have three columns, and we want to avoid invoking the
;; attribute via org, so making this.

(loop for cols in '("1" "2" "3" "4" "5")
  do (loop for rule in '("solid" "none")
    do (eval (read (concat
"(defun org-special-block-extras--" cols "parallel"
(if (equal rule "solid") "" "NB")
"(backend contents)"
"(format (pcase backend"
"('html \"<div style=\\\\"column-rule-style:\" rule \";column-count:\" cols \";\\\\"%s</div>\")"
"('latex \\"\\\\\\par \\\\'setlength{\\\\\\columnseprule}{\" (if (equal rule "solid") "2" "0") "pt}"
"
\\\\\\begin{minipage}[t]{\\\\\\linewidth}"
"
\\\\\\begin{multicols}{\" cols \"}"
"
%s"
"
\\\\\\end{multicols}\\\\\\end{minipage}\\'')) contents))\"))))))

(defalias #'org-special-block-extras--parallel #'org-special-block-extras--2parallel)
(defalias #'org-special-block-extras--parallelNB #'org-special-block-extras--2parallelNB)

```

( The Emacs Web Wowser, M-x eww, does not display parallel environments as desired. )