Package ‘emphatic’

May 28, 2024

**Type** Package

**Title** Exploratory Analysis of Tabular Data using Colour Highlighting

**Version** 0.1.8

**Maintainer** Mike Cheng <mikefc@coolbutuseless.com>

**Description** Tools for exploratory analysis of tabular data using colour highlighting. Highlighting is displayed in any console supporting 'ANSI' colours, and can be converted to 'HTML', 'typst', 'latex' and 'SVG'. 'quarto' and 'rmarkdown' rendering are directly supported. It is also possible to add colour to regular expression matches and highlight differences between two arbitrary R objects.

**URL** [https://coolbutuseless.github.io/package/emphatic/](https://coolbutuseless.github.io/package/emphatic/),
[https://github.com/coolbutuseless/emphatic](https://github.com/coolbutuseless/emphatic)

**BugReports** [https://github.com/coolbutuseless/emphatic/issues](https://github.com/coolbutuseless/emphatic/issues)

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.3.1

**Depends** R (>= 2.10)

**Suggests** knitr, rmarkdown, dplyr, ggplot2, tidyr, purrr, testthat, openxlsx

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Mike Cheng [aut, cre, cph]

**Repository** CRAN

**Date/Publication** 2024-05-28 16:10:05 UTC

1
R topics documented:

- as.character.emphatic .................................................. 2
- as_emphatic ................................................................. 3
- as_html ................................................................. 4
- as_latex ............................................................. 5
- as_svg ................................................................. 5
- as_svgAnim ......................................................... 6
- as_svg_group ...................................................... 7
- as_ttypst .............................................................. 8
- challenger .......................................................... 9
- hl ................................................................. 10
- hl_adjust .......................................................... 11
- hl_diff ........................................................... 12
- hl_grep ............................................................ 13
- hl_opts ............................................................ 15
- is_emphatic .......................................................... 16
- knit_print.emphatic .................................................. 16
- print.emphatic ...................................................... 17
- sea_ice_area ......................................................... 17
- show_html .......................................................... 18
- sydney_rain ......................................................... 18
- write_xlsx .......................................................... 19

Index 20

---

as.character.emphatic  Convert an emphatic data.frame, matrix or atomic vector into a character string.

Description

The output contains ANSI escape codes to colour the elements in the object. This string would then be suitable to pass on to fansi for further manipulation e.g. conversion to HTML for displaying in a vignette.

Usage

```r
## S3 method for class 'emphatic'
as.character(x, ..., mode = "ansi")
```

Arguments

- `x`  
  emphatic data.frame, matrix or atomic vector
- `...`  
  other arguments passed on to `format()`
- `mode`  
  Render mode 'ansi' (default) or 'html' determines how the colours will be represented in text. If you're in a terminal or console, then choose 'ansi'.

---
as_emphatic

Value

A character string of the requested mode

Examples

```
mtcars |> as_emphatic() |> as.character()
```

as_emphatic

Convert a data.frame, matrix or atomic vector into an emphatic version

Description

This usually does not need to be called explicitly by the user.

Usage

```
as_emphatic(.data)
```

Arguments

```
.data data.frame, matrix or atomic vector
```

Details

The function adds the attributes necessary for keeping track of the colours assigned to each cell. This consists of 2 character matrices - one for the text colour and one for the background colour.

Colour information is stored as R colour names (e.g. 'red') or 6 character hex colours (e.g. '#ff0000').

Value

An emphatic version of the given .data with added attributes for text and fill colours

Examples

```
mtcars |> head() |> as_emphatic()
```
as_html

Render an emphatic object to HTML

Description

Render an emphatic object to HTML.

Usage

```r
as_html(
  x,
  ..., 
  font_size = NULL,
  style = list(),
  complete = FALSE,
  browsable = FALSE
)
```

Arguments

- `x`: emphatic object
- `...`: other arguments passed to `as.character.emphatic()`
- `font_size`: CSS font-size. Default: NULL means to not adjust font size. Otherwise, use valid CSS `font-size` specification e.g. "3em", "22px" etc.
- `style`: html tag styling to apply to the `<pre>` wrapper for the returned HTML
- `complete`: logical. Default: FALSE. If TRUE, then add DOCTYPE and the tags for 'html', 'body' and 'head' to make a complete standalone html file.
- `browsable`: Should the SVG be rendered to the RStudio Viewer pane when printed (instead of console output)? Default: FALSE

Value

Character string containing HTML representation

Examples

```r
hl_diff('hello', 'there') |> 
  as_html() |> 
  cat()
```
as_latex

Render an emphatic object to Latex

Description

Render an emphatic object to Latex

Usage

as_latex(x, ..., font_size = NULL)

Arguments

x
emphatic object
...
other arguments passed to as.character.emphatic()
font_size
Integer value indicating font size measured in points. Default: NULL.

Value

single character string containing a latex representation

Examples

hl_diff("hello", "there") |>
  as_latex() |>
  cat()

as_svg

Wrap a single emphatic object into an SVG for display

Description

This is mainly useful within a github README.md since github will not rendered html-styled text in colour, but will render it correctly if it is within a <svg> tags.

Usage

as_svg(
  x,
  width = 1200,
  height = 900,
  ..., 
  font_size = NULL,
  style = list(),
  browsable = FALSE
)
Arguments

- `x`: emphatic object
- `width, height`: viewBox dimensions for SVG
- `...`: other arguments passed to `as.character.emphatic()`
- `font_size`: CSS font-size. Default: NULL means to not adjust font size. Otherwise, use valid CSS font-size specification e.g. "3em", "22px" etc.
- `style`: html tag styling to apply to the `<pre>` wrapper for the returned HTML
- `browsable`: Should the SVG be rendered to the RStudio Viewer pane when when printed (instead of console output)? Default: FALSE

Details

This is just a the results of `as_html()` wrapped in `<svg>` tags

Value

Character string containing SVG representation

Examples

```r
hl_diff('hello', 'there') |> as_svg() |> cat()
```

`as_svg_anim`  
Wrap multiple emphatic object into an SVG animation

Description

Idea borrowed from pointblank

Usage

```r
as_svg_anim(
  x,
  width = 1200,
  height = 900,
  duration = 1,
  playback = c("infinite", "click"),
  font_size = NULL,
  style = list(),
  svg_id = NULL,
  browsable = FALSE
)
```
as_svg_group

Arguments

- **x**: list of emphatic objects
- **width, height**: viewBox dimensions for SVG
- **duration**: frame duration in seconds. May be a single value used for all frames, or a vector of values (one duration value for each frame). Can be fractions of a second.
- **playback**: 'click', 'infinite'
- **font_size**: CSS font-size. Default: NULL means to not adjust font size. Otherwise, use valid CSS font-size specification e.g. "3em", "22px" etc.
- **style**: html tag styling to apply to the <pre> wrapper for the returned HTML
- **svg_id**: ID to use for the SVG tag. Default: NULL means to create a random ID
- **browsable**: Should the SVG be rendered to the RStudio Viewer pane when when printed (instead of console output)? Default: FALSE

Value

Character string containing an animated SVG representation displaying all elements sequentially

Examples

```r
list(
  hl_diff('hello', 'there'),
  hl_diff('goodbye', 'good boy')
) |> as_svg_anim() |> cat()
```

Description

This function wraps html in SVG group tags (i.e. <g>). This may then be wrapped in <svg> tags to create a stand-alone SVG.

Usage

```r
as_svg_group(
  x, 
  width = 1200,  
  height = 900,  
  font_size = NULL,  
  style = list(),  
  visible = TRUE, 
  extra = NULL, 
  ...
)
```
as_typst


description

render an emphatic object to typst

Usage

as_typst(x, ..., font_size = 10, font = NA, line_spacing = 0.3)

Arguments

x  emphatic object
... other arguments passed to as.character.emphatic()
font_size  font size in points. default: 10
font  name of font. Default: NA means to just use the default raw font
line_spacing  line spacing in em units. Default: 0.3
challenger

Value

Character string containing typst representation

Examples

   hl_diff("hello", "there") |>
   as_typst() |>
   cat()

challenger             Challenger o-ring dataset

Description

A dataset containing information about the o-ring status of the flights leading up to the Space Shuttle Challenger disaster.

Usage

challenger

Format

A data.frame

flight  Flight number
temp    Launch temperature (Fahrenheit)
erosion Number of o-ring erosion incidents
blowby  Number of o-ring blow-by incidents
damage  Damage severity index
date    Date of launch

Details

Sourced from a table in Tufte’s "Visual and Statistical Thinking"
Highlight elements in a data.frame

Description

Highlight elements in a data.frame by specifying rows and columns, and the colour to be applied. The colour can be either a vector of colours expressed as characters (e.g. 'red', '#ff0000'), or a ggplot2 Scale object e.g. scale_colour_viridis_c().

Usage

hl(
  .data,
  palette,
  rows = NULL,
  cols = NULL,
  scale_apply,
  elem = "fill",
  show_legend = FALSE,
  opts = hl_opts()
)

Arguments

.data emphatic data.frame
palette  colours to use for highlighting. This may be a single R colour, a vector of R colours, or a ggplot2 style "Scale" object e.g. scale_colour_continuous().
rows, cols specification for rows and columns to target. Default is NULL for both rows and columns, which will target all columns/rows. When palette argument is a scale object, then cols indicates the columns which will be used to calculate the extents of the scale.
scale_apply Only valid when palette is a scale object, specify the target columns to colour. If missing (the default), this function will only colour the column specified in the cols argument. Use NULL to colour all columns.
elem  Apply the highlighting to the 'fill' (the background) or the 'text'. Default: 'fill'
show_legend if a scale object is used for colour, and show_legend = TRUE, then a colourbar legend will be added to the bottom of the output. Default: FALSE
opts create options list

Value

An emphatic object suitable to output to console (for example)
Row and Column Specifications

Specifying rows and columns can be done in a number of ways. These methods are similar to the ideas of tidyselect and dplyr commands such as filter() and select()

numeric vector  row or column indices specified as a numeric vector e.g. c(1, 2, 8)
character vector vector of names matching row or column names e.g. c('mpg', 'wt')
vector of symbols/names vector of symbols which will be evaluated as column names e.g. c(mpg, wt)
numeric range range of indices specified using the : operator e.g. 1:8
symbolic range range of columns specified using the : operator e.g. mpg:wt
tidyselect-style selectors starts_with(), ends_with(), everything(), all_of(), any_of(), matches() contains(), row_number(), n(). These work similar to dplyr and tidyselect but are bespoke implementations so there may be some differences

NULL specifying NULL means that all rows/columns will be selected
all() specifying all() means that all rows/columns will be selected
code that will evaluate to row positions For row selection only, the user can specify code which will evaluate to a logical vector of rows which the highlighting should apply to. These will look like statements used in dplyr::filter(). E.g. cyl == 6 & mpg > 20

Examples

# Simple
mtcars |> head() |> hl(c('red', 'blue'))

# More involved example
mtcars |> head() |> hl(ggplot2::scale_colour_viridis_c(),
  rows = cyl == 6,
  cols = mpg,
  scale_apply = c(mpg, cyl)
)

hl_adjust

Set options for printing on the emphatic matrix or data.frame

Description

Set options for printing on the emphatic matrix or data.frame

Usage

hl_adjust(.data, na, full_colour, text_mode, text_contrast)
hl_diff

Colour the differences between character representations of objects

Description

Highlight the differences between two strings in terms of substitutions, insertions and deletions calculated by the generalized Levenshtein (edit) distance (using adist())

Usage

hl_diff(
  x,
  y,
  coerce = "default",
  fill = NULL,
  text = NULL,
  opts = hl_opts(),
  sep = NULL,
  ...
)
hl_grep

Arguments

- **x, y**: each argument is a single string. Vectors of strings not currently supported.
- **coerce**: How should non-character arguments be coerced to character strings?
  - **default**: the given object x must already be a character string
  - **character**: performs the matching after first calling `as.character(x)`
  - **print**: performs the matching against the default `print(x)` output
  - **deparse**: performs the matching after first calling `deparse1(x)`
  - **str**: performs the matching on the output of calling `str(x)`
- **fill**: named list of colours for substitutions, insertions and deletions with names 'sub', 'ins' and 'del'. If set to NULL (the default) then default colours will be used.
- **text**: named list of colours for the text for 'sub', 'ins' and 'del' operations. If NULL, then colours which contrast with fill will be chosen automatically
- **opts**: create options list
- **sep**: character string of the line separating the two objects. Default: NULL for no separation. Use the empty string to insert an empty line.
- **...**: further arguments passed to adist()

Details

This works character-by-character, so the displayed difference for multiline strings can be quite busy if there are a lot of changes.

Value

- list of 'emphatic' objects which could be rendered to ANSI (for example)

Examples

```r
hl_diff('hello', 'there')
```

hl_grep

Colour highlighting a regular expression search

Description

Highlight text within an R object which matches a given regex. This only works in a terminal which supports ANSI colour codes.

There are slightly different versions of the highlighting function depending upon which text version of the object you’d like to match against:
Usage

hl_grep(
  x,
  pattern,
  coerce = "default",
  opts = hl_opts(),
  fill = NULL,
  text = NULL,
  ...,  
  perl = TRUE
)

Arguments

x             character string
pattern       regular expression string. Note: don’t get too fancy here
coerce        How should non-character arguments be coerced to character strings?
              default - the given object x must already be a character string
              character - performs the matching after first calling as.character(x)
              print - performs the matching against the default print(x) output
              deparse - performs the matching after first calling deparse1(x)
              str - performs the matching on the output of calling str(x)
opts          create options list
fill          solid colour for background. If NULL (the default), then the default colour will be selected
text          text colour. If NULL (the default), then a colour will be seleted which contrasts with the fill colour.
...            extra args passed to gsub
perl          logical. use perl style regex. default: TRUE

Value

An emphatic object suitable to output to console (for example)

Examples

hl_grep(mode, 'switch')
Create a set of options

Usage

hl_opts()

Arguments

- **na**: Character string to display for NA values. Default 'NA'
- **full_colour**: Use 24bit ANSI escape codes? default: FALSE - use 8bit colour. Note: RStudio only supports 8 bit ANSI output (24bit ANSI is rendered invisibly in R studio). For 24bit colour output, try R in the terminal e.g. 'iTerm' on OSX.
- **text_mode**: How to handle text if no text colour has been explicitly specified by the user.
  - **contrast** (default) automatically select a contrasting colour for enhanced readability.
  - **asis** render text in the default text colour for the output device, unless the user has already specified a text colour at this location
  - **remove** remove all text without a user-defined colour
- **text_contrast**: When text_mode='contrast' this numeric value in range [0, 1] adjusts the visibility. Default: 1 (high contrast)

Value

named list of standard options

Examples

# Generate a standard set of options
hl_opts()
is_emphatic

Check if data.frame, matrix or atomic vector is a valid emphatic version

Description

Check if data.frame, matrix or atomic vector is a valid emphatic version

Usage

is_emphatic(x)

Arguments

x Object to test

Value

Logical value

Examples

mtcars |> hl('red') |> is_emphatic()

knit_print.emphatic

Automatically output emphatic objects to HTML knitted documents.

Description

Automatically output emphatic objects to HTML knitted documents.

Usage

knit_print.emphatic(x, style = list(), ...)

Arguments

x emphatic object

style html tag styling to apply to the <pre> wrapper for the returned HTML

... other arguments passed to as.character.emphatic()

Value

a character vector suitable for output during an rmarkdown render
print.emphatic

Examples

mtcars |> 
  hl('red') |> 
  knit_print.emphatic()

print.emphatic  Print an emphatic data.frame, matrix or atomic vector

Description

Print an emphatic data.frame, matrix or atomic vector

Usage

## S3 method for class 'emphatic'
print(x, ...)

Arguments

x  emphatic data.frame, matrix or atomic vector

... other arguments passed on to format()

Value

None.

Examples

mtcars |> 
  head() |> 
  hl('red') |> 
  print()

sea_ice_area  Monthly Southern Sea Ice Area over the last 40 years

Description

From the 'National Snow and Ice Data Center' https://nsidc.org/data/g02135

Usage

sea_ice_area

Format

Matrix of sea ice area, monthly from 1978 to 2020.
**show_html**

*Show HTML or SVG content in the rstudio viewer pane*

**Description**

Show HTML or SVG content in the rstudio viewer pane

**Usage**

```r
show_html(x, viewer = getOption("viewer", utils::browseURL))
```

**Arguments**

- `x` : svg or html
- `viewer` : function which activates viewer

**Value**

None

**Examples**

```r
# This example will try and spawn an external viewer for HTML content
hl_grep(mode, "switch") |> as_html() |> show_html()
```

---

**sydney_rain**

*Monthly total rainfall in Centennial Park, Sydney, Australia*

**Description**

From the Australian Bureau of Meteorology

**Usage**

```r
sydney_rain
```

**Format**

data.frame with each row representing a year, and each column representing a month of that year
write_xlsx

Write an emphatic data.frame to an Excel workbook

Description

Requires openxlsx package

Usage

write_xlsx(x, xlsx_filename, colNames = TRUE, opts = hl_opts())

Arguments

x         emphatic data.frame object
xlsx_filename xlsx filename
colNames  Display column names? logical. Default: TRUE
opts      rendering options

Value

None

Examples

mtcars |>
  hl('blue') |>
  write_xlsx(tempfile())
Index

* datasets
  challenger, 9
  sea_ice_area, 17
  sydney_rain, 18

as.character.emphatic, 2
as_emphatic, 3
as_html, 4
as_latex, 5
as_svg, 5
as_svg_anim, 6
as_svg_group, 7
as_typst, 8

challenger, 9

hl, 10
hl_adjust, 11
hl_diff, 12
hl_grep, 13
hl_opts, 15

is_emphatic, 16

knit_print.emphatic, 16

print.emphatic, 17

sea_ice_area, 17
show_html, 18
sydney_rain, 18

write_xlsx, 19