Package 'highlightHTML'

January 27, 2018

Type Package

Title Highlight HTML Text and Tables

Version 0.2.1

Maintainer Brandon LeBeau <lebebr01+highlightHTML@gmail.com>

Description A tool to format R markdown with CSS ids for HTML output. The tool may be most helpful for those using markdown to create reproducible documents. The biggest limitations in formatting is the knowledge of CSS by the document authors.

Depends R (>= 3.0.0)

Suggests shiny, testthat, dplyr, knitr, rmarkdown

License MIT + file LICENSE

RoxygenNote 6.0.1

VignetteBuilder knitr

URL https://github.com/lebebr01/highlightHTML

BugReports https://github.com/lebebr01/highlightHTML/issues

NeedsCompilation no

Author Brandon LeBeau [aut, cre] (<https://orcid.org/0000-0002-1265-8761>)

Repository CRAN

Date/Publication 2018-01-24 20:55:14

R topics documented:

highlight_html *Master highlight HTML function*

Description

This function inputs a markdown or rmarkdown document and exports an HTML file. The HTML file is then processed to search for tags that inject CSS automatically into the HTML file.

Usage

```
highlight_html(input, output, tags, browse = TRUE, print = FALSE,
  render = TRUE)
```

Arguments

input	File name of markdown or rmarkdown file to highlight the cells of the table or text. Alternatively, if render = FALSE, a HTML file can be specified as the input.
output	Output file name of highlighted HTML file
tags	character vector with CSS tags to be added
browse	logical, If TRUE (default) output file opens in default browser, if FALSE, file is written, but not opened in browser.
print	logical, if TRUE print output to R console, if false (default) output is filtered to other methods (see browse or output).
render	logical, if TRUE (default) will call the rmarkdown::render() function to convert Rmd or md files to html prior to injecting CSS.

Details

A function that allows the alteration of HTML using CSS. This may be helpful coming from a markdown or R markdown file to alter aspects of the page based on a specific criteria. This function handles both tables as well as normal text. The options are only limited based on your knowledge of CSS.

Examples

```
# Example of simple test table
# Change background color of table cells
library(highlightHTML)

# Setting path for example html files
# To see path where these are saved, type file or file1 in the
# r console.
file <- system.file('examples', 'bgtable.html', package = 'highlightHTML')

# Creating CSS tags to inject into HTML document
```

```
tags <- c("#bgred {background-color: #FF0000;}",
  "#bgblue {background-color: #0000FF;}")

# Command to post-process HTML file - Writes to temporary file
highlight_html(input = file, output = tempfile(fileext = ".html"),
  tags = tags, browse = FALSE)

# Change background color and text color with CSS
tags <- c("#bgred {background-color: #FF0000; color: white;}",
  "#bgblue {background-color: #0000FF; color: white;}")

# Post-process HTML file
highlight_html(input = file, output = tempfile(fileext = ".html"),
  tags = tags, update_css = TRUE, browse = TRUE, print = FALSE)

# By default the new file is opened in your default browser, here set to FALSE
highlight_html(input = file, output = tempfile(fileext = ".html"),
  tags = tags, browse = FALSE, print = FALSE)

# Setting path for example html files
# To see path where these are saved, type file or file1 in the
# r console.
file <- system.file('examples', 'bgtext.html', package = 'highlightHTML')

# Change background color and text color with CSS
tags <- c("#bgblack {background-color: black; color: white;}",
  "#bgblue {background-color: #0000FF; color: white;}",
  "#colgreen {color: green;}")

# Post-process HTML file
highlight_html(input = file, output = tempfile(fileext = ".html"),
  tags = tags, browse = TRUE)

# Use of render
file <- system.file('examples', 'mwe.md', package = 'highlightHTML')
tags <- c("#bgred {background-color: #FF0000; color: white;}",
   "#bgblue {background-color: #0000FF; color: white;}",
   "#bgblack {background-color: #000000; color: white;}",
   "#colgold {color: #FFD700;}")
highlight_html(input = file, output = tempfile(fileext = '.html'),
  tags = tags, browse = TRUE, render = TRUE)
```

`print.highlightHTML` *Prints highlightHTML object*

Description

Prints highlightHTML object

Usage

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

Arguments

x	An object from highlightHTML function.
...	Additional arguments passed to function.

---

rgb2hex *Convert RGB to hex*

Description

Enter a list of RGB color codes, or R colors, and get the appropriate hex color code.

Usage

```
rgb2hex(rgbcode = NULL, rcolor = NULL)
```

Arguments

rgbcode	List of rgb color codes, each list must be a vector of three objects representing the three components of rgb color code from 0 - 255. This can be a named list where the name represents the name of the color to be used.
rcolor	An unnamed list of R color names.

Examples

```
rgb2hex(rcolor = list("sienna2", "thistle1"))
rgb2hex(rcolor = list("sienna2", "thistle1"), rgbcode = list('orange' = c(238, 74, 24),
'raw umber' = c(113, 75, 35)))
rgb2hex(rgbcode = list('orange' = c(238, 74, 24), 'raw umber' = c(113, 75, 35)))
```

---

shinyRGB2hex *Run shiny app*

Description

Function that automatically opens shiny app to convert rgb codes to hexidecimal codes

Usage

```
shinyRGB2hex(...)
```

Arguments

...	Other arguments to pass, currently does nothing.

table_id_inject *Table hash addition for markup*

Description

A helper function to include a hashtag id code within a summary table. The summary table most commonly will take the form of a data frame object. For example, a descriptive summary table coming from the `summarise` function from the dplyr package. Can also specify a count table using the `table` function.

Usage

```
table_id_inject(table, id, conditions, variable = NULL, num_digits = NULL)
```

Arguments

`table`	A summary table object, most commonly will be a data.frame, but can also be a count table using the `table` function.
`id`	A vector of css id(s) to include
`conditions`	A character vector of conditions to include id. Must be same length as id. See details and examples for more information on how to specify the conditions.
`variable`	An optional list of column names to specify search of conditions. More than one variable can be specified in each element of the list. The list must be the same length as the conditions or id arguments.
`num_digits`	A numeric value to specify the number of decimal values to include in the final output.

Details

The `conditions` argument takes the following operators for numeric variables: `>`, `>=`, `<`, `<=`, `==`. For character variables, only `==` can be used to specify the text string to match on. Care needs to be made to wrap ensure the text string is wrapped in quotations. See the examples for more details on this.

This function can also be part of a chain using the `%>%` operator from magrittr. See the examples for more details.

Examples

```
library(dplyr)
library(highlightHTML)

mtcars %>%
  group_by(cyl) %>%
  summarise(avg_mpg = mean(mpg), sd_mpg = sd(mpg)) %>%
  data.frame() %>%
  table_id_inject(id = c('#bgred', '#bgblue', '#bggreen'),
```

```
     conditions = c('< 2', '> 16', '== 15.1'))


mtcars %>%
  group_by(cyl) %>%
  summarise(avg_mpg = mean(mpg), sd_mpg = sd(mpg)) %>%
  data.frame() %>%
  table_id_inject(id = c('#bgred', '#bgblue'),
     conditions = c('<= 2', '< 16'),
     variable = list(c('sd_mpg'), c('avg_mpg')))

# text example
storms %>%
  group_by(status) %>%
  summarise(avg_wind = mean(wind)) %>%
  data.frame() %>%
  table_id_inject(id = c('#bgred'),
     conditions = c('== "tropical depression"'))


# Table object
table(mtcars$cyl, mtcars$disp) %>%
  table_id_inject(id = c('#bgred'),
                  conditions = c('>= 3'))
```

Index