HTML in GT table - mtcars

Andrew vanderWilden

11/11/2020

```
library(kableExtra)
library(gt)
library(sparkline)
library(glue)
library(espnscrapeR)
library(tidyverse)
```

```
df <- mtcars
```

```
mpg_list <- split(df$mpg, df$cyl)

mpg_list
```

```
## $`4`
##  [1] 22.8 24.4 22.8 32.4 30.4 33.9 21.5 27.3 26.0 30.4 21.4
##
## $`6`
## [1] 21.0 21.0 21.4 18.1 19.2 17.8 19.7
##
## $`8`
##  [1] 18.7 14.3 16.4 17.3 15.2 10.4 10.4 14.7 15.5 15.2 13.3 19.2 15.8 15.0
```

```
# KableExtra for boxplot in table
data.frame(
  cyl = c(4,6,8),
  boxplot = ''
) %>%
  kbl(booktabs = TRUE) %>%
  kable_paper(full_width = FALSE) %>%
  column_spec(2, image = spec_boxplot(mpg_list, width = 300, height = 70))
```

cyl	boxplot
4	
6	
8	

NOTE: THIS ONLY WORKS WITH EXTERNAL OBJECT, NOT INFO IN DATAFRAME ITSELF

```
# gt and KableExtra for sparklines

mpg_range <- range(df$mpg)

df %>%
  group_by(cyl) %>%
  summarise(data = list(mpg), .groups = 'drop') %>%
  mutate(
    # spec_plot creates plot in base R and returns as svg or pdf
    plot = map(data, ~spec_plot(.x, ylim = mpg_range,
                                same_lim = TRUE, width = 300, height = 70)),
    plot = map(plot, 'svg_text'),
    # need to let gt know to treat svg_text as html
    plot = map(plot, gt::html)
  ) %>%
  select(-data) %>%
  gt()
```

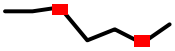
cyl	plot
4	
6	
8	

```
# Do it all in gt

# custom function
gt_plot <- function(table_data, column, plot_data, plot_fun, ...){
  text_transform(
    table_data,
    # note the use of {{}} here - this is tidy eval
    # that allows you to indicate specific columns
    locations = cells_body(columns = vars({{column}})),
    fn = function(x){
      plot <- map(plot_data, plot_fun, width = 300, height = 70, same_lim = T, ...)
      plot_svg <- map(plot, 'svg_text')
      map(plot_svg, gt::html)
    }
  )
}


tibble(cyl = c(4,6,8), boxplot = '') %>%
  gt() %>%
  gt_plot(
    column = boxplot, # column to create boxplot in
    plot_data = mpg_list, # external data to reference
    plot_fun = spec_boxplot, # which plot function
    lim = mpg_range # range applied
  )
```

cyl	boxplot
4	
6	
8	

```
# can quickly change to sparkline instead of boxplot

tibble(cyl = c(4,6,8), sparkline = '') %>%
  gt() %>%
  gt_plot(
    column = sparkline, # column to create plot in
    plot_data = mpg_list, # external data to reference
    plot_fun = spec_plot, # which plot function
    lim = mpg_range, # range applied
    col = 'black', # color of sparkline
    cex = 8 # change size of points
  )
```

cyl	sparkline

cyl	sparkline
4	
6	
8	

```
# put all of them together

#Note that we are just varying the ylim on the line/points 1 vs 2, where the
#mpg_line1/mpg_points1 share a common y-axis, and line2/points2 have their own y-axis.

tibble(
  cyl = c(4,6,8),
  boxplot = '',
  mpg_hist = '',
  mpg_line1 = '',
  mpg_line2 = '',
  mpg_points1 = '',
  mpg_points2 = '',
  mpg_poly = ''
) %>%
  gt() %>%
  gt_plot(column = boxplot, plot_data = mpg_list,
          plot_fun = spec_boxplot, lim = mpg_range) %>%
  gt_plot(column = mpg_hist, plot_data = mpg_list,
          plot_fun = spec_hist, lim = mpg_range) %>%
  gt_plot(column = mpg_line1, plot_data = mpg_list,
          plot_fun = spec_plot, ylim = mpg_range) %>%
  gt_plot(column = mpg_line2, plot_data = mpg_list,
          plot_fun = spec_plot) %>%
  gt_plot(column = mpg_points1, plot_data = mpg_list,
          plot_fun = spec_plot, type = 'p', ylim = mpg_range, cex = 4) %>%
  gt_plot(column = mpg_points2, plot_data = mpg_list,
          plot_fun = spec_plot, type = 'p', cex = 4) %>%
  gt_plot(column = mpg_poly, plot_data = mpg_list,
          plot_fun = spec_plot, polymin = 5, ylim = mpg_range)
```

cyl	boxplot	mpg_hist	mpg_line1	mpg_line2	mpg_points1	mpg_points2	mpg_
4							
6							
8							

Use purrr::pluck() to reference actual dataframe

```
df %>%
  group_by(cyl) %>%
  summarise(mpg_data = list(mpg), .groups = 'drop') %>%
  gt() %>%
  text_transform(
    locations = cells_body(columns = vars(mpg_data)),
    fn = function(x){
      # _data is data in gt table, see str() of gt object
      data_in = pluck(., '_data', 'mpg_data')
      plot = map(data_in, ~spec_plot(.x, ylim = mpg_range, same_lim = T,
                                     width = 300, height = 70))
      plot = map_chr(plot, 'svg_text')
    }
  )
```

cyl	mpg_data
4	
6	
8	

```
# alternative without going into gt object

df %>%
  group_by(cyl) %>%
  summarise(mpg_data = list(as.double(mpg)), .groups = 'drop') %>%
  gt() %>%
  text_transform(
    locations = cells_body(columns = vars(mpg_data)),
    fn = function(x){
      # split the strings at each comma
      split_data <- str_split(x, ', ')
      # convert to type double
      data <- map(split_data, as.double)
      # create the plot
      plot <- map(data, ~spec_plot(.x, ylim = mpg_range, same_lim = T,
                                   width = 300, height = 70))
      # extract the svg item
      map(plot, 'svg_text')
    }
  )
```

cyl	mpg_data
4	

cyl	mpg_data
6	
8	

Rewrite gt_plot() function so it works with a dataframe

```
gt_plot <- function(table_data, plot_col, data_col, plot_fun, ...){
  # save the data extract ahead of time
  # to be used in our anonymous function below
  data_in = pluck(table_data, '_data', data_col)

  text_transform(
    table_data,
    # not the use of {{}} here - this is tidy eval
    # that allow you to indicate specific columns
    locations = cells_body(columns = vars({{plot_col}})),
    fn = function(x){
      plot <- map(data_in, plot_fun, width = 300, height = 70, same_lim = F, ...)
      plot_svg <- map(plot, 'svg_text')
      map(plot_svg, gt::html)
    }
  )
}
```

Now it works with grouped list data columns

```
df %>%
  group_by(cyl) %>%
  summarise(mpg_data = list(mpg), .groups = 'drop') %>%
  gt() %>%
  # note you can leave mpg_data unquoted for tidy eval
  # but have to quote mpg_data for the pluck
  gt_plot(mpg_data, 'mpg_data', plot_fun = spec_plot)
```

cyl	mpg_data
4	
6	
8	

Interactive sparkline with javascript

```
tibble(
  var = c("mpg", "wt"),
  sparkline1 = "",
  sparkline2 = "",
  box = ""
) %>%
  gt() %>%
  text_transform(
    locations = cells_body(vars(sparkline1)),
    fn = function(x){
      sparkline <- map(list(mtcars$mpg, mtcars$wt), ~spk_chr(values = .x, chartRangeMin = 0))
      map(sparkline, gt::html)
    }
  ) %>%
  text_transform(
    locations = cells_body(vars(sparkline2)),
    fn = function(x){
      sparkline <- map(list(mtcars$mpg, mtcars$wt), ~spk_chr(values = .x, type = "bar", chartRan
geMin = 0))
      map(sparkline, gt::html)
    }
  ) %>%
  text_transform(
    locations = cells_body(vars(box)),
    fn = function(x){
      sparkline <- map(list(mtcars$mpg, mtcars$wt), ~spk_chr(values = .x, type = "box", chartRan
geMin = 0))
      map(sparkline, gt::html)
    }
  )
```

var	sparkline1	sparkline2	box
mpg			
wt			

Sparkline function for gt()

```
gt_spark <- function(table_data, plot_col, data_col){
  # save the data extract ahead of time
  # to be used in our anonymous function below
  data_in = pluck(table_data, "_data", data_col)

  text_transform(
    table_data,
    # note use of {{}}, same as above for tidy eval
    locations = cells_body(columns = vars({{plot_col}})),
    fn = function(x){
      sparkline_plot <- map(
        data_in,
        ~spk_chr(values = .x, chartRangeMin = 0)
      )

      map(sparkline_plot, gt::html)
    }
  )
}
```

```
# Apply sparkline function in gt table

df %>%
  group_by(cyl) %>%
  summarise(mpg_data = list(mpg), .groups = 'drop') %>%
  gt() %>%
  gt_spark(mpg_data, 'mpg_data')
```

cyl	mpg_data
4	
6	
8	

gt Example with simulated data for medical trial

```
coef_table %>%
  select(-coefficients, -contains('conf')) %>%
  mutate(
    image = map(image, 'svg_text'),
    image = map(image, ~gt::html(as.character(.x)))
  ) %>%
  select(group:Placebo, pct_diff = image) %>%
  gt(
    groupname_col = 'group',
    rowname_col = 'subgroup'
  ) %>%
  opt_row_striping() %>%
  tab_options(
    data_row.padding = px(3)
  )
```

	Inclisiran	Placebo	pct_diff
All Patients	781	780	
Sex			
Male	535	548	
Female	246	232	
Age			
<65 yr	297	333	
>= 65 yr	484	447	
<75 yr	638	649	
>=75 yr	143	131	
Body-Mass index			
<=Median	394	385	
>Median	387	394	
Race			
White	653	685	
Black	110	87	
Other	18	8	
Baseline statin treatment			
Yes	701	692	
No	80	88	

	Inclisiran	Placebo	pct_diff
Intensity of statin treatment			
High	538	546	
Not high	243	234	
Metabolic disease			
Diabetes	371	331	
Metabolic syndrome	195	207	
Neither	215	242	
Renal function			
Normal	395	410	
Mild impairment	269	260	
Moderate impairment	113	107	

Final mtcars table

```
set.seed(377)

df %>%
  tibble() %>%
  select(1:4) %>%
  sample_n(size = 6) %>%
  mutate(
    rank_change = sample(c('increase', 'decrease'), size = 6, replace = T),
    rank_change = map(rank_change, rank_chg)
  ) %>%
  mutate(
    rating = sample(1:5, size = 6, replace = T),
    rating = map(rating, rating_stars)
  ) %>%
  mutate(
    cylinder = map(cyl, add_cyl_color)
  ) %>%
  mutate(
    mpg_plot = mpg/max(mpg) * 100,
    mpg_plot = map(mpg_plot, ~bar_chart(value = .x, color = 'lightblue'))
  ) %>%
  gt() %>%
  cols_align(
    align = 'left',
    columns = vars(mpg_plot)
  ) %>%
  cols_label(
    mpg = gt::html(as.character(with_tooltip('MPG', 'Miles per Gallon')))
  ) %>%
  tab_source_note(
    source_note = html(
      htmltools::tags$a(
        href = 'https://gt.rstudio.com/reference/md.html',
        target = '_blank',
        'Data Source'
      ) %>%
        as.character()
    )
  ) %>%
  tab_source_note(
    source_note = html(
      "<details><h3 style='font-face:bold'>Table Key</h3><div>MPG: Miles Per Gallon</div><div>Cy
l: Cylinders</div><div>disp: Displacement</div><div>hp: Horsepower</div><div>rank_change: Rank C
hange</div><div>rating: Rating</div></details>"

    )
  ) %>%
  tab_options(
    data_row.padding = px(5)
  )
```

MPG (Miles per Gallon)	cyl	disp	hp	rank_change	rating	cylinder	mpg_plot

MPG (Miles per Gallon)	cyl	disp	hp	rank_change	rating	cylinder	mpg_plot
15.5	8	318.0	150	↓	★★★★☆	8 Cylinders	
14.3	8	360.0	245	↓	★☆☆☆☆	8 Cylinders	
21.0	6	160.0	110	↓	★★☆☆☆	6 Cylinders	
10.4	8	472.0	205	↑	★★★★★	8 Cylinders	
26.0	4	120.3	91	↑	★★★★☆	4 Cylinders	
15.0	8	301.0	335	↓	★★☆☆☆	8 Cylinders	

Data Source (https://gt.rstudio.com/reference/md.html)

▶ Details

Second Example with NFL QBs

```
# use espnscrapeR to get NFL standings + QBR ratings
nfl_qbr <- get_nfl_qbr(2020)
nfl_standings <- get_nfl_standings(2020)

# also get weekly for embedded plot
qbr_weekly <- crossing(season = 2020, week = 1:8) %>%
  pmap_dfr(.f = get_nfl_qbr)
```

```
# Data prep

qbr_match <- qbr_weekly %>%
  filter(short_name %in% nfl_qbr$short_name) %>%
  group_by(short_name, team) %>%
  summarise(qbr_weekly = list(qbr_total), .groups = "drop",
            qbr = mean(qbr_total),
            qbr_sd = sd(qbr_total),
            plays = sum(qb_plays),
            pass = mean(pass),
            run = mean(run),
            head = unique(headshot_href),
            n = n()) %>%
  arrange(desc(qbr)) %>%
  filter(n >= 7)

# clean up the data a bit and combine
tab_df <- qbr_match %>%
  left_join(nfl_standings, by = c("team" = "abb_name")) %>%
  select(short_name, team, head, qbr_weekly:run, wins, losses, points_for) %>%
  mutate(wl = glue("{wins}-{losses}")) %>%
  select(-wins, -losses)
tab_df
```

```
## # A tibble: 23 x 11
##    short_name team  head  qbr_weekly   qbr qbr_sd plays  pass   run points_for
##    <chr>      <chr> <chr> <list>     <dbl>  <dbl> <dbl> <dbl> <dbl>      <dbl>
##  1 R. Wilson  SEA   http~ <dbl [7]>   81.2   7.17   328  6.09 1.17         274
##  2 D. Brees   NO    http~ <dbl [7]>   78.7  11.0    280  5.69 0.171        244
##  3 P. Mahomes KC    http~ <dbl [8]>   77.8  17.3    347  6.65 1.29         286
##  4 A. Rodgers GB    http~ <dbl [7]>   77.7  28.2    285  5.89 0.586        253
##  5 J. Allen   BUF   http~ <dbl [8]>   75.0  18.4    362  5.91 1.21         242
##  6 R. Tanneh~ TEN   http~ <dbl [7]>   74.6  17.6    284  5.19 1.13         232
##  7 D. Carr    LV    http~ <dbl [7]>   72.4  17.1    284  4.91 0.357        218
##  8 M. Ryan    ATL   http~ <dbl [8]>   71    26.0    370  4.6  0.662        243
##  9 K. Murray  ARI   http~ <dbl [7]>   69.5  16.3    340  3.41 2.69         234
## 10 D. Watson  HOU   http~ <dbl [7]>   68.3  19.0    305  4.16 0.186        193
## # ... with 13 more rows, and 1 more variable: wl <glue>
```

```
# calc rank change
qbr_rnk_chg <- qbr_weekly %>%
  mutate(game_week = as.integer(game_week)) %>%
  group_by(short_name) %>%
  mutate(mean_qbr = mean(qbr_total)) %>%
  ungroup() %>%
  select(game_week, rank, short_name, qbr_total, mean_qbr) %>%
  filter(game_week != max(game_week)) %>%
  filter(short_name %in% nfl_qbr$short_name) %>%
  group_by(short_name) %>%
  summarize(prev_qbr = mean(qbr_total), mean_qbr = unique(mean_qbr)) %>%
  mutate(
    prev_week = rank(-prev_qbr),
    rank = rank(-mean_qbr)
    ) %>%
  mutate(rank_chg = prev_week-rank) %>%
  ungroup() %>%
  arrange(desc(mean_qbr)) %>%
  select(short_name, qbr = mean_qbr, rank_chg, rank)

qbr_rnk_chg
```

```
## # A tibble: 31 x 4
##    short_name       qbr rank_chg  rank
##    <chr>          <dbl>    <dbl> <dbl>
##  1 R. Wilson       81.2        0     1
##  2 D. Brees        78.7        2     2
##  3 P. Mahomes      77.8        4     3
##  4 A. Rodgers      77.7        1     4
##  5 D. Prescott     75.7        3     5
##  6 J. Allen        75.0       -4     6
##  7 R. Tannehill    74.6       -4     7
##  8 J. Herbert      74.5        2     8
##  9 D. Carr         72.4       -3     9
## 10 R. Fitzpatrick  71.7        1    10
## # ... with 21 more rows
```

```
# Code for name, team, record combo

combine_word <- function(name, team, wl){
      glue::glue(
        "<div style='line-height:10px'><span style='font-weight:bold;font-variant:small-caps;fon
t-size:14px'>{name}</div>
        <div style='line-height:12px'><span style ='font-weight:bold;color:grey;font-size:10px'>
{team}  {wl}</span></div>"
      )
    }

combo_df <- tab_df %>%
  left_join(qbr_rnk_chg, by = c("short_name", "qbr")) %>%
  select(rank, rank_chg, short_name:wl) %>%
  mutate(
    rank = row_number(),
    combo = combine_word(short_name, team, wl),
    combo = map(combo, gt::html)
    ) %>%
  select(rank, rank_chg, head, combo, qbr, qbr_weekly, plays, points_for)
```

```
final_table <- combo_df %>%
  gt() %>%
  cols_align(
    align = 'left',
    columns = vars(combo)
  ) %>%
  tab_options(
    data_row.padding = px(2)
  ) %>%
  text_transform(
    locations = cells_body(columns = vars(head)),
    fn = function(x){
      gt::web_image(x)
    }
  ) %>%
    text_transform(
    locations = cells_body(columns = vars(rank_chg)),
    fn = function(x){

      rank_chg <- as.integer(x)

      choose_logo <-function(x){
        if (x == 0){
        gt::html(fontawesome::fa("equals", fill = "grey"))
      } else if (x > 0){
         gt::html(glue::glue("<span style='color:#1134A6;font-face:bold;font-size:10px;'>{x}</sp
an>"), fontawesome::fa("arrow-up", fill = "#1134A6"))
      } else if (x < 0) {
        gt::html(glue::glue("<span style='color:#DA2A2A;font-face:bold;font-size:10px;'>{x}</spa
n>"), fontawesome::fa("arrow-down", fill = "#DA2A2A"))
      }
      }

      map(rank_chg, choose_logo)

    }
  ) %>%
  fmt_number(
    columns = vars(qbr),
    decimals = 1
    ) %>%
  tab_style(
    style = cell_text(weight = "bold"),
    locations = cells_column_labels(TRUE)
  ) %>%
  cols_label(
    rank = "RK",
    combo = "",
    head = "QB",
    qbr = "QBR",
    plays = "PLAYS",
    points_for = "PF",
    qbr_weekly = "WEEKLY",
```

```
    rank_chg = ""
  ) %>%
  gt_spark(qbr_weekly, "qbr_weekly") %>%
  espnscrapeR::gt_theme_espn() %>%
  tab_source_note(
    source_note = gt::html(
      htmltools::tags$a(
        href = "https://www.espn.com/nfl/qbr",
        target = "_blank",
        "Data: ESPN"
        ) %>%
        as.character()
      )
    ) %>%
  cols_align(
    "left",
    columns = vars(qbr_weekly)
  ) %>%
  cols_width(
    vars(rank) ~ px(25),
    vars(rank_chg) ~ px(35),
    vars(head) ~ px(50),
    vars(combo) ~ px(115),
    vars(qbr) ~ px(35),
    vars(plays) ~ px(50),
    vars(points_for) ~ px(35),
    vars(qbr_weekly) ~ px(75)
  ) %>%
  tab_header(
    title = gt::html("<h3>NFL QBR through Week 8</h3>")
  ) %>%
  tab_options(
    table.width = px(480),
    data_row.padding = px(4)
  )
final_table
```

NFL QBR through Week 8

RK		QB	QBR	WEEKLY	PLAYS	PF
1	=	 **R. WILSON** SEA 6-2	81.2		328	274
2	2 ↑	 **D. BREES** NO 6-2	78.7		280	244
3	4 ↑	 **P. MAHOMES** KC 8-1	77.8		347	286
4	1 ↑	 **A. RODGERS** GB 6-2	77.7		285	253

Data: ESPN (https://www.espn.com/nfl/qbr)

NFL QBR through Week 8

5	-4 ↓		J. Allen BUF 7-2	75.0		362	242
6	-4 ↓		R. Tannehill TEN 6-2	74.6		284	232
7	-3 ↓		D. Carr LV 5-3	72.4		284	218
8	3 ↑		M. Ryan ATL 3-6	71.0		370	243
9	1 ↑		K. Murray ARI 5-3	69.5		340	234
10	2 ↑		D. Watson HOU 2-6	68.3		305	193
11	5 ↑		T. Brady TB 6-3	67.8		349	250
12	=		T. Bridgewater CAR 3-6	65.7		329	210
13	-4 ↓		M. Stafford DET 3-5	65.2		295	197
14	4 ↑		P. Rivers IND 5-3	64.6		256	208
15	-1 ↓		B. Mayfield CLE 5-3	63.9		271	206
16	-10 ↓		L. Jackson BAL 6-2	63.7		282	227
17	=		D. Jones NYG 2-7	62.2		346	168
18	1 ↑		B. Roethlisberger PIT 8-0	60.0		283	235
19	-4 ↓		J. Goff LAR 5-3	59.7		328	193
20	=		J. Burrow CIN 2-5	56.8		426	194
21	3 ↑		K. Cousins MIN 3-5	54.1		232	217
22	=		G. Minshew II JAX 1-7	51.8		331	179
23	-3 ↓		C. Wentz PHI 3-4	50.2		407	186

Data: ESPN (https://www.espn.com/nfl/qbr)