The Family of *Export Functions*

There may come a day when you need to export a **gt** table to some specific format. Saving the table in one of many formats is possible with the `gtsave()` function. We have functions for getting the HTML content of a **gt** table (`as_raw_html()`), getting LaTeX code (`as_latex()`), and getting rich text (`as_rtf()`). Did you use the `summary_rows()` function and wish you had that summary data in a tibble? You can get it out with `extract_summary()` .

`gtsave()` : Save a **gt** table as a file

The `gtsave()` function makes it easy to save a **gt** table to a file. The function guesses the file type by the extension provided in the output filename, producing either an HTML, PDF, PNG, LaTeX, or RTF file.

EXAMPLES

Use `gtcars` to create a *gt** table. Add a stubhead label to describe what is in the stub.

```
tab_1 <-
  gtcars %>%
  dplyr::select(model, year, hp, trq) %>%
  dplyr::slice(1:5) %>%
  gt(rowname_col = "model") %>%
  tab_stubhead(label = "car")
```

Get an HTML file with inlined CSS (which is necessary for including the table as part of an HTML email).

```
# tab_1 %>% gtsave("tab_1.html", inline_css = TRUE)
```

By using `inline_css = FALSE` , we get a more conventional HTML fragment with embedded CSS styles.

```
# tab_1 %>% gtsave("tab_1.html", inline_css = FALSE)
```

Save the HTML table as a PDF file. @e can optionally add a separate `path` .

```
# tab_1 %>% gtsave("tab_1.pdf", path = "~")
```

Saving as PNG file results in a cropped image of an HTML table; the overall size and amount of whitespace can both be set.

```
# tab_1 %>% gtsave("tab_1.png", zoom = 2.5, expand = 10)
```

`as_raw_html()` : Get the HTML content of a **gt** table

Get the HTML content from a `gt_tbl` object as a single-element character vector. By default, the generated HTML will have inlined styles, where CSS styles (that were previously contained in CSS rule sets external to the `<table>` element) are included as style attributes in the HTML table's tags. This option is preferable when using the output HTML table in an emailing context.

EXAMPLES

Use `gtcars` to create a **gt** table. Add a header and then export as HTML code with CSS inlined.

```
tab_html <-
  gtcars %>%
  dplyr::select(mfr, model, msrp) %>%
  dplyr::slice(1:5) %>%
  gt() %>%
  tab_header(
    title = md("Data listing from **gtcars**"),
    subtitle = md("`gtcars` is an R dataset")
  ) %>%
  as_raw_html(inline_css = TRUE)
```

`tab_html` is a single-element vector containing inlined HTML for the table. It has only the `<table>...</table>` part so it's not a complete HTML document but rather an HTML fragment.

```
tab_html %>%
  substr(1, 700) %>%
  cat()
```

```
## <table style="font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen,
Ubuntu, Cantarell, 'Helvetica Neue', 'Fira Sans', 'Droid Sans', Arial, sans-serif; display: t
able; border-collapse: collapse; margin-left: auto; margin-right: auto; color: #333333; font-
size: 16px; font-weight: normal; font-style: normal; background-color: #FFFFFF; width: auto;
border-top-style: solid; border-top-width: 2px; border-top-color: #A8A8A8; border-right-styl
e: none; border-right-width: 2px; border-right-color: #D3D3D3; border-bottom-style: solid; bo
rder-bottom-width: 2px; border-bottom-color: #A8A8A8; border-left-style: none; border-left-wi
dth: 2px; border-left-color: #D3D3D3;">
##   <thead styl
```

`as_latex()`: Output a **gt** object as LaTeX

Get the LaTeX content from a `gt_tbl` object as a `knit_asis` object. This object contains the LaTeX code and attributes that serve as LaTeX dependencies (i.e., the LaTeX packages required for the table). Using `as.character()` on the created object will result in a single-element vector containing the LaTeX code.

EXAMPLES

Use `gtcars` to create a **gt** table. Add a header and then export as an object with LaTeX code.

```
tab_latex <-
  gtcars %>%
  dplyr::select(mfr, model, msrp) %>%
  dplyr::slice(1:5) %>%
  gt() %>%
  tab_header(
    title = md("Data listing from **gtcars**"),
    subtitle = md("`gtcars` is an R dataset")
  ) %>%
  as_latex()
```

`tab_latex` is a `knit_asis` object, which makes it easy to include in R Markdown documents that are knit to PDF; we can use `as.character()` to get just the LaTeX code as a single-element vector.

```
tab_latex %>%
  as.character() %>%
  cat()
```

```
## \captionsetup[table]{labelformat=empty,skip=1pt}
## \begin{longtable}{llr}
## \caption*{
## \large Data listing from \textbf{gtcars}\\
## \small \texttt{gtcars} is an R dataset\\
## } \\
## \toprule
## mfr & model & msrp \\
## \midrule
## Ford & GT & 447000 \\
## Ferrari & 458 Speciale & 291744 \\
## Ferrari & 458 Spider & 263553 \\
## Ferrari & 458 Italia & 233509 \\
## Ferrari & 488 GTB & 245400 \\
## \bottomrule
## \end{longtable}
```

`extract_summary()` : Extract a summary list from a **gt** object

Get a list of summary row data frames from a `gt_tbl` object where summary rows were added via the `summary_rows()` function. The output data frames contain the `groupname` and `rowname` columns, whereby `rowname` contains descriptive stub labels for the summary rows.

EXAMPLES

Use `sp500` to create a **gt** table with row groups. Create summary rows by row group (`min` , `max` , `avg`) and then extract the summary rows as a list object.

```
summary_extracted <-
  sp500 %>%
  dplyr::filter(date >= "2015-01-05" & date <="2015-01-30") %>%
  dplyr::arrange(date) %>%
  dplyr::mutate(week = paste0("W", strftime(date, format = "%V"))) %>%
  dplyr::select(-adj_close, -volume) %>%
  gt(rowname_col = "date", groupname_col = "week") %>%
  summary_rows(
    groups = TRUE,
    columns = c(open, high, low, close),
    fns = list(
      min = ~min(.),
      max = ~max(.),
      avg = ~mean(.)
    ),
    formatter = fmt_number,
    use_seps = FALSE
  ) %>%
  extract_summary()

summary_extracted
```

```
## $summary_df_data_list
## $summary_df_data_list$W02
## # A tibble: 3 x 8
##   group_id rowname  date  open  high   low close  week
##   <chr>    <chr>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 W02      min        NA 2006. 2030. 1992. 2003.    NA
## 2 W02      max        NA 2063. 2064. 2038. 2062.    NA
## 3 W02      avg        NA 2035. 2049. 2017. 2031.    NA
##
## $summary_df_data_list$W03
## # A tibble: 3 x 8
##   group_id rowname  date  open  high   low close  week
##   <chr>    <chr>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 W03      min        NA 1992. 2018. 1988. 1993.    NA
## 2 W03      max        NA 2046. 2057. 2023. 2028.    NA
## 3 W03      avg        NA 2020. 2033. 2000. 2015.    NA
##
## $summary_df_data_list$W04
## # A tibble: 3 x 8
##   group_id rowname  date  open  high   low close  week
##   <chr>    <chr>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 W04      min        NA 2020. 2029. 2004. 2023.    NA
## 2 W04      max        NA 2063. 2065. 2051. 2063.    NA
## 3 W04      avg        NA 2035. 2049. 2023. 2042.    NA
##
## $summary_df_data_list$W05
## # A tibble: 3 x 8
##   group_id rowname  date  open  high   low close  week
##   <chr>    <chr>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 W05      min        NA 2002. 2023. 1989. 1995.    NA
## 2 W05      max        NA 2050. 2058. 2041. 2057.    NA
## 3 W05      avg        NA 2030. 2039. 2009. 2021.    NA
```

Use the summary list to make a new **gt** table. The key thing is to use `dplyr::bind_rows()` and then pass the tibble to `gt()` (the `groupname` and `rowname` columns can be used to create row groups and a stub).

```
summary_extracted %>%
  unlist(recursive = FALSE) %>%
  dplyr::bind_rows() %>%
  gt(rowname_col = "rowname", groupname_col = "groupname") %>%
  cols_hide(columns = c(date, week))
```

	group_id	open	high	low	close
min	W02	2005.550	2029.610	1992.440	2002.610
max	W02	2063.450	2064.430	2038.330	2062.140
avg	W02	2035.240	2048.562	2016.854	2031.208
min	W03	1992.250	2018.400	1988.120	1992.670
max	W03	2046.130	2056.930	2022.580	2028.260

	group_id	open	high	low	close
avg	W03	2020.422	2033.288	1999.772	2014.930
min	W04	2020.190	2028.940	2004.490	2022.550
max	W04	2062.980	2064.620	2050.540	2063.150
avg	W04	2034.557	2048.707	2023.362	2042.410
min	W05	2002.450	2023.320	1989.180	1994.990
max	W05	2050.420	2057.620	2040.970	2057.090
avg	W05	2030.484	2039.186	2008.986	2021.008