The Family of *Image Functions*

We can add images into a gt table with the help of the `*_image()` functions. Two common ways to do this: (1) use `text_transform()` to insert images into data cells, (2) use any function that creates new labels (e.g., `tab_header()`) and use a `*_image()` function within the `html()` helper.

`web_image()`: Helper function for adding an image from the web

We can flexibly add a web image inside of a table with `web_image()` function. The function provides a convenient way to generate an HTML fragment with an image URL. Because this function is currently HTML-based, it is only useful for HTML table output. To use this function inside of data cells, it is recommended that the `text_transform()` function is used. With that function, we can specify which data cells to target and then include a `web_image()` call within the required user-defined function (for the `fn` argument). If we want to include an image in other places (e.g., in the header, within footnote text, etc.) we need to use `web_image()` within the `html()` helper function.

EXAMPLES

Get the PNG-based logo for the R Project from an image URL.

```
r_png_url <-"https://www.r-project.org/logo/Rlogo.png"
```

Create a tibble that contains heights of an image in pixels (one column as a string, the other as numerical values), then, create a **gt** table; use the `text_transform()` function to insert the R logo PNG image with the various sizes.

```
dplyr::tibble(
  pixels = px(seq(10, 35, 5)),
  image = seq(10, 35, 5)
) %>%
  gt() %>%
  text_transform(
    locations = cells_body(columns = image),
    fn = function(x) {
      web_image(
        url = r_png_url,
        height = as.numeric(x)
      )
    }
  )
```

pixels	image
10px	
15px	
20px	
25px	
30px	

pixels	image
35px	

Get the SVG-based logo for the R Project from an image URL.

```
r_svg_url <- "https://www.r-project.org/logo/Rlogo.svg"
```

Create a tibble that contains heights of an image in pixels (one column as a string, the other as numerical values), then, create a **gt** table. Use the `tab_header()` function to insert the R logo SVG image once in the title and five times in the subtitle.

```
dplyr::tibble(
  pixels = px(seq(10, 35, 5)),
  image = seq(10, 35, 5)
) %>%
  gt() %>%
  tab_header(
    title = html(
      "<strong>R Logo</strong>",
      web_image(
        url = r_svg_url,
        height = px(50)
      )
    ),
    subtitle = html(
      web_image(
        url = r_svg_url,
        height = px(12)
      ) %>%
        rep(5)
    )
  )
```

R Logo

pixels	image
10px	10
15px	15
20px	20
25px	25
30px	30
35px	35

`local_image()`: Helper function for adding a local image

We can flexibly add a local image (i.e., an image residing on disk) inside of a table with the `local_image()` function. The function provides a convenient way to generate an HTML fragment using an on-disk PNG or SVG. Because this function is currently HTML-based, it is only useful for HTML table output. To use this function inside of data cells, it is recommended that the `text_transform()` function is used. With that function, we can specify which data cells to target and then include a `local_image()` call within the required user-defined function (for the `fn` argument). If we want to include an image in other places (e.g., in the header, within footnote text, etc.) we need to use `local_image()` within the `html()` helper function.

EXAMPLE

Create a tibble that contains heights of an image in pixels (one column as a string, the other as numerical values), then, create a **gt** table. Use the `text_transform()` function to insert a local test image (PNG) image with the various sizes.

```
dplyr::tibble(
  pixels = px(seq(10, 35, 5)),
  image = seq(10, 35, 5)
) %>%
  gt() %>%
  text_transform(
    locations = cells_body(columns = image),
    fn = function(x) {
      local_image(
        filename = test_image(type = "png"),
        height = as.numeric(x)
      )
    }
  )
```

pixels	image
10px	
15px	
20px	
25px	
30px	
35px	

`ggplot_image()`: Helper function for adding a ggplot

We can add a **ggplot2** plot inside of a table with the help of the `ggplot_image()` function. The function provides a convenient way to generate an HTML fragment with a `ggplot` object. Because this function is currently HTML-based, it is only useful for HTML table output. To use this function inside of data cells, it is recommended that the `text_transform()` function is used. With that function, we can specify which data cells to target and then include a call to `ggplot_image()` within the required user-defined function (for the `fn` argument). If we want to include a plot in other places (e.g., in the header, within footnote text, etc.) we need to use `ggplot_image()` within the `html()` helper function.

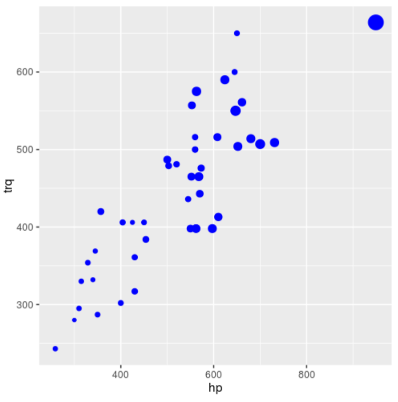
EXAMPLE

Create a ggplot plot.

```
plot_object <-
  ggplot(data = gtcars, aes(x = hp, y = trq, size = msrp)) +
  geom_point(color = "blue") +
  theme(legend.position = "none")
```

Create a tibble that contains two cells (where one is a placeholder for an image), then, create a **gt** table. Use the `text_transform()` function to insert the plot using by calling `ggplot_image()` within the user-defined function.

```
dplyr::tibble(
  text = "Here is a ggplot:",
  ggplot = NA
) %>%
  gt() %>%
  text_transform(
    locations = cells_body(columns = ggplot),
    fn = function(x) {
      plot_object %>%
        ggplot_image(height = px(200))
    }
  )
```

text	ggplot
Here is a ggplot:	

`test_image()`: Generate a path to a test image

Two test images are available within the **gt** package. Both contain the same imagery (sized at 200px by 200px) but one is a PNG file while the other is an SVG file. This function is most useful when paired with `local_image()` since we can test various sizes of the test image within that function.