tt.R

vincent

2024-01-10

Draw a table

@param x A data frame @param output “markdown”, “latex”, or “html”. If `output` is `NULL`, then: * “html” if `knitr::is_html_output()` is `TRUE` * “latex” if `knitr::is_latex_output()` is `TRUE` * Otherwise determined by setting a global option: `options(tt_output_default = "markdown")` @param latex Options to customize LaTeX tables. See `?options_tabularray` and the examples section below. @param html Options to customize HTML tables. See `?options_bootstrap` and the examples section below. @template tabularray @export

```
tt <- function(x,
               output = NULL,
               align = NULL,
               caption = NULL,
               theme = "booktabs",
               extendable = FALSE) {

  # sanity checks
  output <- sanitize_output(output)
  assert_data_frame(x)
  assert_string(caption, null.ok = TRUE)
  assert_string(align, null.ok = TRUE)

  # build table
  if (output == "latex") {
    out <- tinytable_tabularray(x, caption = caption, theme = theme, extendable = extendable)

  } else if (output == "html"){
    out <- tinytable_html(x, caption = caption, settings = options)

  } else {
    out <- tinytable_markdown(x, caption = caption)
  }


  if (!is.null(align)) {
    if (nchar(align) != ncol(x)) {
      msg <- sprintf("`align` must have length %s, equal to the number of columns in `x`.", ncol(x))
      stop(msg, call. = FALSE)
    }
    align <- strsplit(align, split = "")[[1]]
    if (!all(align %in% c("l", "c", "r"))) {
      msg <- "Elements of `align` must be 'c', 'l', or 'r'."
      stop(msg, call. = FALSE)
```

```
    }
    for (j in seq_along(align)) {
      out <- style_tt(out, j = j, align = align[[j]])
    }
  }

  return(out)
}
```