December 31, 2014

0.0.1 genNavbar

`genNavbar` generates a navigation bar for a web page. The html file created is generally written to the project's `docs/Rmd/include` directory. However, if this function is used to create a navbar for a Github user web page, the html file should be stored in a sensible location inside the user pages repository, e.g., `leonawicz.github.io/assets`.

The common navigation bar html is included at the beginning of the body of the html for each web page in the project's website. `menu` is a vector of names for each dropdown menu. `submenus` is a list of vectors of menu options corresponding to each menu. `files` is a similar list of vectors. Each element is either an html file for a web page to be associated with the submenu link, "header" to indicate the corresponding name in `submenus` is only a group label and not a link to a web page, or "divider" to indicate placement of a bar for separating groups in a dropdown menu.

`theme` can be either `united` (default) or `cyborg`. Both are from Bootswatch. The function must apply some internal differentiation in the construction of the html navigation bar between themes. This is currently the only `rpm` function which attempts to handle multiple Bootswatch themes with different CSS tags.

```
# Functions for Github project websites
genNavbar <- function(htmlfile = "navbar.html", title, menu, submenus, files,
    title.url = "index.html", home.url = "index.html", site.url = "", site.name = "Github",
    theme = "united", include.home = FALSE) {
    if (!(theme %in% c("united", "cyborg")))
        stop("Only the following themes supported: united, cyborg.")

    navClassStrings <- function(x) {
        switch(x, united = c("brand", "nav-collapse collapse", "nav", "nav pull-right",
            "navbar-inner", "container", "", "btn btn-navbar", ".nav-collapse",
            "</div>\n"), cyborg = c("navbar-brand", "navbar-collapse collapse navbar-responsive-collapse
            "nav navbar-nav", "nav navbar-nav navbar-right", "container", "navbar-header",
            "      </div>\n", "navbar-toggle", ".nav-collapse", ""))
    }

    ncs <- navClassStrings(theme)

    fillSubmenu <- function(x, name, file, theme) {
        if (theme == "united")
            dd.menu.header <- "nav-header" else if (theme == "cyborg")
            dd.menu.header <- "dropdown-header"
        if (file[x] == "divider")
            return("              <li class=\"divider\"></li>\n")
        if (file[x] == "header")
            return(paste0("              <li class=\"", dd.menu.header, "\">",
                name[x], "</li>\n"))
        paste0("              <li><a href=\"", file[x], "\">", name[x], "</a></li>\n")
    }

    fillMenu <- function(x, menu, submenus, files, theme) {
```

```
        m <- menu[x]
        gs.menu <- gsub(" ", "-", tolower(m))
        s <- submenus[[x]]
        f <- files[[x]]
        if (s[1] == "empty") {
            y <- paste0("<li><a href=\"", f, "\">", m, "</a></li>\n")
        } else {
            y <- paste0("<li class=\"dropdown\">\n            <a href=\"", gs.menu,
                "\" class=\"dropdown-toggle\" data-toggle=\"dropdown\">", m,
                " <b class=\"caret\"></b></a>\n            <ul class=\"dropdown-menu\">\n",
                paste(sapply(1:length(s), fillSubmenu, name = s, file = f, theme = theme),
                  sep = "", collapse = ""), "            </ul>\n", collapse = "")
        }
    }

    if (include.home)
        home <- paste0("<li><a href=\"", home.url, "\">Home</a></li>\n          ") else home <- ""
    x <- paste0("<div class=\"navbar navbar-default navbar-fixed-top\">\n  <div class=\"",
        ncs[5], "\">\n    <div class=\"", ncs[6], "\">\n      <button type=\"button\" class=\"",
        ncs[8], "\" data-toggle=\"collapse\" data-target=\"", ncs[9], "\">\n        <span class=\"icon-b
        ncs[1], "\" href=\"", title.url, "\">", title, "</a>\n", ncs[7], "      <div class=\"",
        ncs[2], "\">\n        <ul class=\"", ncs[3], "\">\n          ", home,
        paste(sapply(1:length(menu), fillMenu, menu = menu, submenus = submenus,
            files = files, theme = theme), sep = "", collapse = "\n          "),
        "        </ul>\n        <ul class=\"", ncs[4], "\">\n          <a class=\"btn btn-primary\" href
        site.url, "\">\n            <i class=\"fa fa-github fa-lg\"></i>\n            ",
        site.name, "\n          </a>\n        </ul>\n      </div><!--/.nav-collapse -->\n    </div>\n  "
        ncs[10], "</div>\n", collpase = "")
    sink(htmlfile)
    cat(x)
    sink()
    x
}
```

0.0.2 genOutyaml

`genOutyaml` generates the `_out.yaml` file for yaml front-matter common to all html files in the project website. The file should be written to the project's `docs/Rmd` directory. `lib` specifies the library directory for any associated files. yaml `includes` for external html common to all project web pages in the site can also be specified with `header`, `before_body`, and `after_body`. These can be specified by file basename only (no path) and the function assumes these files are in the `docs/Rmd/include` directory. At this time all external libraries must be provided by the user, for example in `docs/Rmd/libs`. It is recommended. See the project repo [gh-pages](https://github.com/leonawicz/ProjectManagement/tree/gh-pages "gh-pages") branch for an example.

```
genOutyaml <- function(file, theme = "cosmo", highlight = "zenburn", lib = NULL,
    header = NULL, before_body = NULL, after_body = NULL) {
    output.yaml <- paste0("html_document:\n  self_contained: false\n  theme: ",
        theme, "\n  highlight: ", highlight, "\n  mathjax: null\n  toc_depth: 2\n")
    if (!is.null(lib))
        output.yaml <- paste0(output.yaml, "  lib_dir: ", lib, "\n")
    output.yaml <- paste0(output.yaml, "  includes:\n")
    if (!is.null(header))
```

```
        output.yaml <- paste0(output.yaml, "    in_header: ", header, "\n")
    if (!is.null(before_body))
        output.yaml <- paste0(output.yaml, "    before_body: ", before_body,
            "\n")
    if (!is.null(after_body))
        output.yaml <- paste0(output.yaml, "    after_body: ", after_body, "\n")
    sink(file)
    cat(output.yaml)
    sink()
    output.yaml
}
```