Package ‘fresh’

June 26, 2024

Title Create Custom 'Bootstrap' Themes to Use in 'Shiny'

Version 0.2.1

Description Customize 'Bootstrap' and 'Bootswatch' themes, like colors, fonts, grid layout, to use in 'Shiny' applications, 'rmarkdown' documents and 'flexdashboard'.

URL `https://github.com/dreamRs/fresh`

BugReports `https://github.com/dreamRs/fresh/issues`

License GPL-3

Encoding UTF-8

Imports bslib,
htmltools,
rstudioapi,
sass,
shiny

Suggests bsicons,
shinyWidgets,
shinydashboard,
bs4Dash,
knitr,
rmarkdown,
testthat (>= 2.1.0),
covr

RoxygenNote 7.3.1

Roxygen list(markdown = TRUE)

VignetteBuilder knitr

R topics documented:

adminlte_color *AdminLTE CSS colors variables*

Description

Those variables can be used to customize defaults colors in shinydashboard.

Usage

```
adminlte_color(
  light_blue = NULL,
  red = NULL,
  green = NULL,
  aqua = NULL,
  yellow = NULL,
  blue = NULL,
  navy = NULL,
  teal = NULL,
  olive = NULL,
  lime = NULL,
  orange = NULL,
  fuchsia = NULL,
  purple = NULL,
  maroon = NULL,
  black = NULL,
  gray_lte = NULL
)
```

Arguments

light_blue	Light blue (primary status), default to #3c8dbc.
red	Red (danger status), default to #dd4b39.
green	Green (success status), default to #00a65a.
aqua	Aqua (info status), default to #00c0ef.
yellow	Yellow (warning status), default to #f39c12.
blue	Blue, default to #0073b7.
navy	Navy, default to #001F3F.
teal	Teal, default to #39CCCC.
olive	Olive, default to #3D9970.
lime	Lime, default to #01FF70.
orange	Orange, default to #FF851B.
fuchsia	Fuchsia, default to #F012BE.
purple	Purple, default to #605ca8.
maroon	Maroon, default to #D81B60.
black	Black, default to #111.
gray_lte	Gray, default to #d2d6de.

Value

a list that can be used in create_theme.

Examples

```
adminlte_color(
  light_blue = "#086A87",
  aqua = "#A9D0F5",
  green = "#0B3B0B",
  purple = "#610B4B"
)

if (interactive()) {
  library(shiny)
  library(shinydashboard)

  ui <- dashboardPage(
    header = dashboardHeader(title = "My dashboard"),
    sidebar = dashboardSidebar(
      sidebarMenu(
        menuItem(
          "Dashboard",
          tabName = "dashboard",
          icon = icon("dashboard")
        )
      )
    ),
    body = dashboardBody(

      use_theme(create_theme(
        adminlte_color(
          light_blue = "#086A87",
          aqua = "#A9D0F5",
          green = "#0B3B0B",
          purple = "#610B4B"
        )
      )),

      tabItems(
        tabItem(
          "dashboard",

          # infoBoxes
          fluidRow(
            infoBox(
              "Orders", uiOutput("orderNum2"),
              "Subtitle", icon = icon("credit-card")
            ),
            infoBox(
              "Approval Rating", "60%",
              icon = icon("line-chart"), color = "green",
              fill = TRUE
            ),
            infoBox(
              "Progress", "20%",
               icon = icon("users"),
               color = "purple"
            )
          ),

          # valueBoxes
          fluidRow(
            valueBox(
              5846, "New Orders",
              icon = icon("credit-card"),
              href = "http://google.com"
            ),
            valueBox(
              tagList("60",
                      tags$sup(style="font-size: 20px", "%")),
              "Approval Rating",
              icon = icon("line-chart"),
               color = "green"
```

```
            ),
            valueBox(
              "42%", "Progress",
              icon = icon("users"),
              color = "purple"
            )

          )
        )
      )
    )
  )

  server <- function(input, output, session) {

  }

  shinyApp(ui, server)
}
```

adminlte_global *AdminLTE CSS global variables*

Description

Those variables can be used to customize global settings in shinydashboard.

Usage

```
adminlte_global(content_bg = NULL, box_bg = NULL, info_box_bg = NULL)
```

Arguments

content_bg	Background color of the body.
box_bg	Default background color for boxes.
info_box_bg	Default background color for info boxes.

Value

a list that can be used in create_theme.

Examples

```
if (interactive()) {
  library(shiny)
  library(shinydashboard)

  ui <- dashboardPage(
    header = dashboardHeader(title = "My dashboard"),
    sidebar = dashboardSidebar(),
    body = dashboardBody(

      use_theme(create_theme(
        adminlte_global(
          content_bg = "#FAAC58"
        )
      ))
    )
  )

  server <- function(input, output, session) {

  }

  shinyApp(ui, server)
}
```

adminlte_sidebar *AdminLTE CSS sidebar variables*

Description

Those variables can be used to customize the sidebar in shinydashboard.

Usage

```
adminlte_sidebar(
  width = NULL,
  dark_bg = NULL,
  dark_hover_bg = NULL,
  dark_color = NULL,
  dark_hover_color = NULL,
  dark_submenu_bg = NULL,
  dark_submenu_color = NULL,
  dark_submenu_hover_color = NULL,
  light_bg = NULL,
  light_hover_bg = NULL,
  light_color = NULL,
  light_hover_color = NULL,
  light_submenu_bg = NULL,
  light_submenu_color = NULL,
  light_submenu_hover_color = NULL
)
```

Arguments

width	Side bar width, default to 230px.

`dark_bg` Background color (dark mode).

`dark_hover_bg` Background hover color (dark mode).

`dark_color` Text color (dark mode).

`dark_hover_color` Text hover color (dark mode).

`dark_submenu_bg` Background sub-menu color (dark mode).

`dark_submenu_color` Text sub-menu color (dark mode).

`dark_submenu_hover_color` Text sub-menu hover color (dark mode).

`light_bg` Background color (light mode).

`light_hover_bg` Background hover color (light mode).

`light_color` Text color (light mode).

`light_hover_color` Text hover color (light mode).

`light_submenu_bg` Background sub-menu color (light mode).

`light_submenu_color` Text sub-menu color (light mode).

`light_submenu_hover_color` Text sub-menu hover color (light mode).

Value

a `list` that can be used in `create_theme`.

Examples

```
if (interactive()) {
  library(shiny)
  library(shinydashboard)

  ui <- dashboardPage(
    header = dashboardHeader(title = "My dashboard"),
    sidebar = dashboardSidebar(
      sidebarMenu(
        menuItem("Dashboard", tabName = "dashboard", icon = icon("dashboard")),
        menuItem("Widgets", icon = icon("th"), tabName = "widgets", badgeLabel = "new",
                 badgeColor = "green"),
        menuItem("Charts", icon = icon("bar-chart-o"),
                 menuSubItem("Sub-item 1", tabName = "subitem1"),
                 menuSubItem("Sub-item 2", tabName = "subitem2")
        )
      )
    ),
    body = dashboardBody(
      use_theme(create_theme(
        adminlte_sidebar(
          dark_bg = "#F5A9A9",
          dark_hover_bg = "#8A0808"
        )
      ))
    )
  )

  server <- function(input, output, session) {

  }

  shinyApp(ui, server)
}
```

`adminlte_vars` *AdminLTE 2 custom variables*

Description

Use any AdminLTE or Bootstrap variables to customize a shinydashboard theme.

Usage

```
adminlte_vars(...)
```

Arguments

`...` Variables to use, under the form `body_bg = "#FFF"` or `"body-bg" = "#FFF"`.

Value

a `list` that can be used in `create_theme`.

Note

For a full list of available variables, use `search_vars_adminlte2`.

Examples

```
adminlte_vars(body_bg = "#FFF")

adminlte_vars("body-bg" = "#FFF")
```

bs-theme-nord *Bootstrap Theme based on Nord palette*

Description

Bootstrap Theme based on Nord palette

Usage

```
bs_theme_nord_light(version = bslib::version_default(), preset = "bootstrap")

bs_theme_nord_dark(version = bslib::version_default(), preset = "bootstrap")
```

Arguments

`version`	The major version of Bootstrap to use, see `bslib::bs_theme()`.
`preset`	The name of a theme preset, default to "bootstrap".

Value

Returns a `sass::sass_bundle()` (list-like) object.

Examples

```
library(shiny)
library(bslib)
library(bsicons)

make_btns <- function(outline = FALSE) {
  lapply(
    X = c("primary", "secondary", "success", "danger", "warning", "info"),
    FUN = function(x) {
      if (outline)
        x <- paste0("outline-", x)
    actionButton(paste0("btn", x), paste("Button", x), class = paste("btn", x, sep = "-"))
    }
  )
}

ui <- page_navbar(
  title = "Nord theme",

  theme = bs_theme_nord_light(),

  fillable = FALSE,
  sidebar = sidebar(
    "Sidebar",
    selectInput("id", "Choice:", month.name)
  ),
  nav_panel(
    "Page 1",

    make_btns(),
    tags$br(), tags$br(),
    make_btns(TRUE),
    tags$br(), tags$br(),

    layout_column_wrap(
      width = 1 / 5,
      value_box(
        title = "1st value",
        value = "123",
        showcase = bs_icon("bar-chart"),
        theme = "primary",
        p("The 1st detail")
      ),
      value_box(
        title = "2nd value",
        value = "456",
        showcase = bs_icon("graph-up"),
        theme = "success",
        p("The 2nd detail"),
        p("The 3rd detail")
      ),
      value_box(
        title = "3rd value",
        value = "789",
        showcase = bs_icon("pie-chart"),
        theme = "danger",
        p("The 4th detail"),
        p("The 5th detail"),
        p("The 6th detail")
      ),
      value_box(
        title = "4th value",
        value = "123",
        showcase = bs_icon("bar-chart"),
        theme = "warning",
        p("The 1st detail")
      ),
      value_box(
        title = "5th value",
        value = "123",
        showcase = bs_icon("bar-chart"),
        theme = "info",
        p("The 1st detail")
      )
    ),
    layout_column_wrap(
      width = 1 / 5,
      card(
        card_header(
          class = "bg-primary",
          "A header"
        ),
        card_body("Some content")
      ),
      card(
        card_header(
          class = "bg-success",
          "A header"
        ),
        card_body("Some content")
      ),
      card(
        card_header(
          class = "bg-danger",
          "A header"
        ),
        card_body("Some content")
      ),
      card(
        card_header(
          class = "bg-warning",
          "A header"
        ),
        card_body("Some content")
      ),
      card(
        card_header(
          class = "bg-info",
          "A header"
        ),
        card_body("Some content")
      )
    )
  ),
  nav_panel("Page 2", "Page 2 content"),
  nav_spacer(),
  nav_item(
    input_switch("dark_mode", "Dark mode", width = "120px")
  )
)

server <- function(input, output, session) {
  observe(session$setCurrentTheme(
    if (isTRUE(input$dark_mode)) bs_theme_nord_dark() else bs_theme_nord_light()
  ))
}

if (interactive())
  shinyApp(ui, server)
```

bs3-variables *Bootstrap custom variables*

Description

Use any Bootstrap variables to customize a shiny theme.

Usage

```
bs_vars(...)

bs_vars_global(
  body_bg = NULL,
  text_color = NULL,
  link_color = NULL,
  link_hover_color = NULL,
  line_height_base = NULL,
  grid_columns = NULL,
  grid_gutter_width = NULL,
  border_radius_base = NULL
)

bs_vars_color(
  brand_primary = NULL,
  brand_success = NULL,
  brand_info = NULL,
  brand_warning = NULL,
  brand_danger = NULL,
  gray_base = NULL,
  gray_darker = NULL,
  gray_dark = NULL,
  gray = NULL,
  gray_light = NULL,
  gray_lighter = NULL
)

bs_vars_navbar(
  height = NULL,
  margin_bottom = NULL,
  border_radius = NULL,
  padding_horizontal = NULL,
  padding_vertical = NULL,
  collapse_max_height = NULL,
  default_color = NULL,
  default_bg = NULL,
  default_border = NULL,
  default_link_color = NULL,
  default_link_active_color = NULL,
  default_link_active_bg = NULL,
  default_link_hover_color = NULL,
  default_link_hover_bg = NULL,
  inverse_color = NULL,
  inverse_bg = NULL,
  inverse_border = NULL,
  inverse_link_color = NULL,
  inverse_link_active_color = NULL,
  inverse_link_active_bg = NULL,
  inverse_link_hover_color = NULL,
  inverse_link_hover_bg = NULL
)

bs_vars_nav(
  link_padding = NULL,
  link_hover_bg = NULL,
  disabled_link_color = NULL,
  disabled_link_hover_color = NULL
)

bs_vars_font(
  family_sans_serif = NULL,
  size_base = NULL,
  size_large = NULL,
  size_small = NULL,
  size_h1 = NULL,
  size_h2 = NULL,
  size_h3 = NULL,
  size_h4 = NULL,
  size_h5 = NULL,
  size_h6 = NULL
)

bs_vars_wells(bg = NULL, border = NULL)

bs_vars_state(
  success_text = NULL,
  success_bg = NULL,
  success_border = NULL,
  info_text = NULL,
  info_bg = NULL,
  info_border = NULL,
  warning_text = NULL,
  warning_bg = NULL,
  warning_border = NULL,
  danger_text = NULL,
  danger_bg = NULL,
  danger_border = NULL
)

bs_vars_input(
  bg = NULL,
  color = NULL,
  border = NULL,
  border_radius = NULL,
  color_placeholder = NULL,
  group_addon_bg = NULL,
  border_focus = NULL,
  bg_disabled = NULL
)

bs_vars_alert(
  padding = NULL,
  border_radius = NULL,
  link_font_weight = NULL,
  success_text = NULL,
  success_bg = NULL,
  success_border = NULL,
  info_text = NULL,
  info_bg = NULL,
  info_border = NULL,
  warning_text = NULL,
  warning_bg = NULL,
  warning_border = NULL,
  danger_text = NULL,
  danger_bg = NULL,
  danger_border = NULL
)

bs_vars_progress(
  bg = NULL,
  bar_color = NULL,
  border_radius = NULL,
  bar_bg = NULL,
  bar_success_bg = NULL,
  bar_warning_bg = NULL,
  bar_danger_bg = NULL,
  bar_info_bg = NULL
)

bs_vars_panel(
  bg = NULL,
  body_padding = NULL,
  heading_padding = NULL,
  footer_padding = NULL,
  border_radius = NULL,
  inner_border = NULL,
  footer_bg = NULL,
  default_text = NULL,
  default_border = NULL,
  default_heading_bg = NULL,
  primary_text = NULL,
  primary_border = NULL,
  primary_heading_bg = NULL,
  success_text = NULL,
  success_border = NULL,
  success_heading_bg = NULL,
  info_text = NULL,
  info_border = NULL,
  info_heading_bg = NULL,
  warning_text = NULL,
  warning_border = NULL,
  warning_heading_bg = NULL,
  danger_text = NULL,
  danger_border = NULL,
  danger_heading_bg = NULL
)

bs_vars_modal(
  md = NULL,
  lg = NULL,
  sm = NULL,
  inner_padding = NULL,
  title_padding = NULL,
  title_line_height = NULL,
  content_bg = NULL,
  content_border_color = NULL,
  content_fallback_border_color = NULL,
  backdrop_bg = NULL,
  backdrop_opacity = NULL,
  header_border_color = NULL,
  footer_border_color = NULL
)

bs_vars_button(
  font_weight = NULL,
  default_color = NULL,
  default_bg = NULL,
  default_border = NULL,
  primary_color = NULL,
  primary_bg = NULL,
  primary_border = NULL,
  success_color = NULL,
  success_bg = NULL,
  success_border = NULL,
  info_color = NULL,
  info_bg = NULL,
  info_border = NULL,
  warning_color = NULL,
  warning_bg = NULL,
  warning_border = NULL,
  danger_color = NULL,
  danger_bg = NULL,
  danger_border = NULL,
  link_disabled_color = NULL,
  border_radius_base = NULL,
  border_radius_large = NULL,
  border_radius_small = NULL
)

bs_vars_dropdown(
  bg = NULL,
  border = NULL,
  fallback_border = NULL,
  divider_bg = NULL,
  link_color = NULL,
  link_hover_color = NULL,
  link_hover_bg = NULL,
  link_active_color = NULL,
  link_active_bg = NULL,
  link_disabled_color = NULL,
  header_color = NULL
)

bs_vars_pills(
  border_radius = NULL,
  active_link_hover_bg = NULL,
  active_link_hover_color = NULL
)

bs_vars_tabs(
  border_color = NULL,
  link_hover_border_color = NULL,
  active_link_hover_bg = NULL,
  active_link_hover_color = NULL,
  active_link_hover_border_color = NULL,
  justified_link_border_color = NULL,
  justified_active_link_border_color = NULL
)

bs_vars_badge(
  color = NULL,
  bg = NULL,
  link_hover_color = NULL,
  active_color = NULL,
  active_bg = NULL,
  font_weight = NULL,
  line_height = NULL,
  border_radius = NULL
)

bs_vars_component(
  padding_base_vertical = NULL,
  padding_base_horizontal = NULL,
  padding_large_vertical = NULL,
  padding_large_horizontal = NULL,
  padding_small_vertical = NULL,
  padding_small_horizontal = NULL,
  padding_xs_vertical = NULL,
  padding_xs_horizontal = NULL,
  line_height_large = NULL,
  line_height_small = NULL,
  border_radius_base = NULL,
  border_radius_large = NULL,
  border_radius_small = NULL,
  component_active_color = NULL,
  component_active_bg = NULL,
  caret_width_base = NULL,
  caret_width_large = NULL
)

bs_vars_table(
  cell_padding = NULL,
  condensed_cell_padding = NULL,
  bg = NULL,
  bg_accent = NULL,
  bg_hover = NULL,
  bg_active = NULL,
  border_color = NULL
)
```

Arguments

...	Variables to use, under the form body_bg = "#FFF" or "body-bg" = "#FFF".
body_bg	Background color for the body.
text_color	Global text color on body.
link_color	Dropdown link text color.
link_hover_color	Linked badge text color on hover.

`line_height_base`
Unit-less `line-height` for use in components like buttons.

`grid_columns` Number of columns in the grid, e.g. in `shiny::fluidRow(shiny::column(...))`.

`grid_gutter_width`
Padding between columns. Gets divided in half for the left and right.

`border_radius_base`
Base border radius.

`brand_primary` Primary color, default: #337ab7.

`brand_success` Success color, default: #5cb85c.

`brand_info` Info color, default: #5bc0de.

`brand_warning` Warning color, default: #f0ad4e.

`brand_danger` Danger color, default: #d9534f.

`gray_base` Base gray color.

`gray_darker` Darker gray color.

`gray_dark` Dark gray color.

`gray` Gray color.

`gray_light` Light gray color.

`gray_lighter` Lighter gray color.

`height` Height of the navbar, e.g. `"50px"` (the default in Bootstrap).

`margin_bottom` Bottom margin of navbar.

`border_radius` Border radius.

`padding_horizontal`
Horizontal padding.

`padding_vertical`
= Vertical padding.

`collapse_max_height`
Max height when collapsed.

`default_color` Text color for default buttons.

`default_bg` Background color for default buttons.

`default_border` Border color for default buttons.

`default_link_color`
Link color.

`default_link_active_color`
Color for active link (selected tab).

`default_link_active_bg`
Background color for active link (selected tab).

`default_link_hover_color`
Color of links when hovered.

`default_link_hover_bg`
Background color of links when hovered.

`inverse_color` Color of text for inverted navbar.

`inverse_bg` Background color for inverted navbar.

`inverse_border` Border color for inverted navbar.

`inverse_link_color`
Link color for inverted navbar.

`inverse_link_active_color`
Color for active link (selected tab) for inverted navbar.

`inverse_link_active_bg`
Background color for active link (selected tab) for inverted navbar.

`inverse_link_hover_color`
Color of links when hovered for inverted navbar.

`inverse_link_hover_bg`
Background color of links when hovered for inverted navbar.

`link_padding` Padding for links (tabset's titles).

`link_hover_bg` Hover background for dropdown links.

`disabled_link_color`
Disabled link color.

`disabled_link_hover_color`
Disabled link hover color.

`family_sans_serif`
Font family to use.

`size_base` Size of base font, e.g. normal text, default in Bootstrap is `"15px"`.

`size_large` Size of large text.

`size_small` Size of small text.

`size_h1` Size of h1 tags.

`size_h2` Size of h2 tags.

`size_h3` Size of h3 tags.

`size_h4` Size of h4 tags.

`size_h5` Size of h5 tags.

`size_h6` Size of h6 tags.

`bg` Background color.

`border` Dropdown menu border-color.

`success_text` Text color for success status.

`success_bg` Background color for success buttons.

`success_border` Border color for success buttons.

`info_text` Text color for info status.

`info_bg` Background color for info buttons.

`info_border` Border color for info buttons.

`warning_text` Text color for warning status.

`warning_bg` Background color for warning buttons.

`warning_border` Border color for warning buttons.

`danger_text`	Text color for danger status.
`danger_bg`	Background color for danger buttons.
`danger_border`	Border color for danger buttons.
`color`	Text color.
`color_placeholder`	Text color of placeholder.
`group_addon_bg`	Background color of addons.
`border_focus`	Color of border when focused.
`bg_disabled`	Background color for disabled input.
`padding`	Padding for alerts.
`link_font_weight`	Font weight for links in alerts.
`bar_color`	Progress bar text color
`bar_bg`	Default progress bar color.
`bar_success_bg`	Success progress bar color.
`bar_warning_bg`	Warning progress bar color.
`bar_danger_bg`	Danger progress bar color.
`bar_info_bg`	Info progress bar color.
`body_padding`	Panel body padding.
`heading_padding`	Panel heading padding.
`footer_padding`	Panel footer padding.
`inner_border`	Border color for inner elements in panel.
`footer_bg`	Panel footer background color.
`default_text`	Default color for text.
`default_heading_bg`	Default background color for panel heading.
`primary_text`	Text color for primary status.
`primary_border`	Border color for primary buttons.
`primary_heading_bg`	Heading background color for primary status.
`success_heading_bg`	Heading background color for success status.
`info_heading_bg`	Heading background color for info status.
`warning_heading_bg`	Heading background color for warning status.
`danger_heading_bg`	Heading background color for danger status.
`md`	Size in pixel for medium modal, e.g. `modalDialog(size = "m")`.

`lg` Size in pixel for large modal, e.g. `modalDialog(size = "l")`.

`sm` Size in pixel for small modal, e.g. `modalDialog(size = "s")`.

`inner_padding` Padding applied to the modal body.

`title_padding` Padding applied to the modal title.

`title_line_height` Modal title line-height.

`content_bg` Background color of modal content area.

`content_border_color` Modal content border color.

`content_fallback_border_color` Modal content border color (for IE8).

`backdrop_bg` Modal backdrop background color.

`backdrop_opacity` Modal backdrop opacity.

`header_border_color` Modal header border color.

`footer_border_color` Modal footer border color.

`font_weight` Font weight, e.g. : `"bold"`.

`primary_color` Text color for primary buttons.

`primary_bg` Background color for primary buttons.

`success_color` Text color for success buttons.

`info_color` Text color for info buttons.

`warning_color` Text color for warning buttons.

`danger_color` Text color for danger buttons.

`link_disabled_color` Disabled dropdown menu item background color.

`border_radius_large` Large border radius.

`border_radius_small` Small border radius.

`fallback_border` Dropdown menu border-color (for IE8).

`divider_bg` Divider color for between dropdown items.

`link_active_color` Active dropdown menu item text color.

`link_active_bg` Active dropdown menu item background color.

`header_color` Text color for headers within dropdown menus.

`active_link_hover_bg` Active link hover background color.

`active_link_hover_color` Active link hover color.

`border_color` Border color.

`link_hover_border_color`
Link hover color.

`active_link_hover_border_color`
Active link hover border color.

`justified_link_border_color`
Justified link border color.

`justified_active_link_border_color`
Justified active link border color.

`active_color` Badge text color in active nav link.

`active_bg` Badge background color in active nav link.

`line_height` Line height.

`padding_base_vertical`
Vertical base padding.

`padding_base_horizontal`
Horizontal base padding.

`padding_large_vertical`
Vertical large padding.

`padding_large_horizontal`
Horizontal large padding.

`padding_small_vertical`
Vertical small padding.

`padding_small_horizontal`
Horizontal small padding.

`padding_xs_vertical`
Vertical extra small padding.

`padding_xs_horizontal`
Horizontal extra small padding.

`line_height_large`
Line height for large elements.

`line_height_small`
Line height for small elements.

`component_active_color`
Color for active components.

`component_active_bg`
Background color for active components.

`caret_width_base`
Width for caret.

`caret_width_large`
Widget for large caret.

`cell_padding` Cell padding.

`condensed_cell_padding`
Cell padding when using condensed table.

`bg_accent` Background color used in striped table.

`bg_hover` Background color used when hovering the table with the mouse.

`bg_active` Background color when row is selected.

Value

a list that can be used in create_theme().

Note

For a full list of available variables, use search_vars_bs().

Examples

```
bs_vars(body_bg = "#FFF")

bs_vars("body-bg" = "#FFF")
```

---

bs4Dash-sidebar *bs4Dash sidebar skins light/dark*

Description

bs4Dash sidebar skins light/dark

Usage

```
bs4dash_sidebar_light(
  bg = NULL,
  hover_bg = NULL,
  color = NULL,
  hover_color = NULL,
  active_color = NULL,
  submenu_bg = NULL,
  submenu_color = NULL,
  submenu_hover_color = NULL,
  submenu_hover_bg = NULL,
  submenu_active_color = NULL,
  submenu_active_bg = NULL,
  header_color = NULL
)

bs4dash_sidebar_dark(
  bg = NULL,
  hover_bg = NULL,
  color = NULL,
  hover_color = NULL,
  active_color = NULL,
  submenu_bg = NULL,
  submenu_color = NULL,
  submenu_hover_color = NULL,
  submenu_hover_bg = NULL,
  submenu_active_color = NULL,
  submenu_active_bg = NULL,
  header_color = NULL
)
```

Arguments

bg	Background color.
hover_bg	Hover background color.
color	Color.
hover_color	Hover color.
active_color	Active color.
submenu_bg	Submenu background color.
submenu_color	Submenu color.
submenu_hover_color	Submenu hover color.
submenu_hover_bg	Submenu hover background color.
submenu_active_color	Submenu active color.
submenu_active_bg	Submenu active background color.
header_color	Header color.

Value

a list that can be used in create_theme.

Examples

```
# Change colors used in bs4Dash
bs4dash_sidebar_light(
  bg = "#D7DF01",
  color = "#FF0000",
  active_color = "#00FF00",
  submenu_bg = "#00FFFF"
)

if (interactive()) {

  library(shiny)
  library(bs4Dash)

  ui <- bs4DashPage(
    title = "bs4Dash Custom Sidebar",
    navbar = bs4DashNavbar(),
    sidebar = bs4DashSidebar(
      title = "bs4Dash Custom Sidebar",
      skin = "light",
      bs4SidebarHeader("Sidebar Title"),
      bs4SidebarMenu(
        bs4SidebarMenuItem(
          tabName = "menu1",
          text = "Menu 1",
          icon = "home"
        ),
        bs4SidebarMenuItem(
          tabName = "menu2",
          text = "Menu 2",
          icon = "th"
        ),
        bs4SidebarMenuItem(
          text = "Item List",
          icon = "bars",
          startExpanded = TRUE,
          bs4SidebarMenuSubItem(
            text = "Item 1",
            tabName = "item1",
            icon = "circle-thin"
          ),
          bs4SidebarMenuSubItem(
            text = "Item 2",
            tabName = "item2",
            icon = "circle-thin"
          )
        )
      )
    ),
    body = bs4DashBody(
      use_theme(create_theme(
        bs4dash_sidebar_light(
          bg = "#D7DF01",
          color = "#FF0000",
          active_color = "#00FF00",
          submenu_bg = "#00FFFF"
        )
      ))
    )
  )

  server <- function(input, output) {

  }

  shinyApp(ui, server)

}
```

bs4dash_button *bs4dash buttons variables*

Description

bs4dash buttons variables

Usage

```
bs4dash_button(
  default_background_color = NULL,
  default_color = NULL,
  default_border_color = NULL,
  padding_y_xs = NULL,
  padding_x_xs = NULL,
  line_height_xs = NULL,
  font_size_xs = NULL,
  border_radius_xs = NULL
)
```

Arguments

default_background_color	Default background color.
default_color	Default color.
default_border_color	Default border color.
padding_y_xs	Vertical padding for extra small button.
padding_x_xs	Horizontal padding for extra small button.
line_height_xs	Line height for extra small button.
font_size_xs	Font size for extra small button.
border_radius_xs	Border radius for extra small button.

Value

a `list` that can be used in `create_theme`.

Examples

```
# This will affect default actionButton()
bs4dash_button(
  default_background_color = "#FF0000",
  default_color = "#3ADF00",
  default_border_color = "#3ADF00"
)
```

```
if (interactive()) {

  library(shiny)
  library(bs4Dash)

  ui <- bs4DashPage(
    title = "bs4Dash Custom Colors",
    navbar = bs4DashNavbar(),
    sidebar = bs4DashSidebar(),
    body = bs4DashBody(

      use_theme(create_theme(
        bs4dash_button(
          default_background_color = "#FF0000",
          default_color = "#3ADF00",
          default_border_color = "#3ADF00"
        )
      )),

      actionButton(
        "btn",
        "An action button",
        icon("rocket")
      )
    )
  )

  server <- function(input, output) {

  }

  shinyApp(ui, server)

}
```

bs4dash_color *bs4Dash main colors*

Description

bs4Dash main colors

Usage

```
bs4dash_color(
  blue = NULL,
  lightblue = NULL,
  navy = NULL,
  cyan = NULL,
  teal = NULL,
  olive = NULL,
  green = NULL,
  lime = NULL,
  orange = NULL,
  yellow = NULL,
  fuchsia = NULL,
  purple = NULL,
  maroon = NULL,
  red = NULL,
  black = NULL,
  gray_x_light = NULL,
  gray_600 = NULL,
  gray_800 = NULL,
  gray_900 = NULL,
  white = NULL
)
```

Arguments

`blue`	Default: #007bff. This color is used for **primary** status.
`lightblue`	Default: #3c8dbc.
`navy`	Default: #001f3f.
`cyan`	Default: #17a2b8. This color is used for **info** status.
`teal`	Default: #39cccc.
`olive`	Default: #3d9970.
`green`	Default: #28a745. This color is used for **success** status.
`lime`	Default: #01ff70.
`orange`	Default: #ff851b.
`yellow`	Default: #ffc107. This color is used for **warning** status.
`fuchsia`	Default: #f012be.
`purple`	Default: #605ca8.
`maroon`	Default: #d81b60.
`red`	Default: #dc3545. This color is used for **danger** status.
`black`	Default: #111.
`gray_x_light`	Default: #d2d6de.
`gray_600`	Default: #6c757d. This color is used for **secondary** status.
`gray_800`	Default: #343a40. Color for dark skin.
`gray_900`	Default: #212529. Color for text in body.
`white`	Default: #ffffff.

Value

a `list` that can be used in `create_theme`.

Examples

```
# Change colors used in bs4Dash
bs4dash_color(
  blue = "#F7FE2E",
  lightblue = "#01DF3A"
)

if (interactive()) {

  library(shiny)
  library(bs4Dash)

  ui <- bs4DashPage(
    title = "bs4Dash Custom Colors",
    # sidebar_collapsed = FALSE,
    navbar = bs4DashNavbar(),
    sidebar = bs4DashSidebar(
      title = "bs4Dash Custom Colors",
      skin = "light",
      bs4SidebarMenu(
        bs4SidebarMenuItem(
          tabName = "tab1",
          text = "UI components"
        )
      )
    ),
    body = bs4DashBody(

      use_theme(create_theme(
        bs4dash_color(
          blue = "#F7FE2E",
          navy = "#01DF3A"
        )
      )),

      bs4TabItems(
        bs4TabItem(
          tabName = "tab1",
          tags$h2("UI components", class = "bg-navy"),
          tags$h4("bs4ValueBox"),
          fluidRow(
            bs4ValueBox(
              value = 150,
              subtitle = "ValueBox with primary status",
              status = "primary",
              icon = "shopping-cart",
              href = "#",
              width = 4
            )
          ),
          tags$h4("bs4Card"),
          fluidRow(
            bs4Card(
              title = "Card with primary status",
              closable = FALSE,
              width = 6,
              solidHeader = TRUE,
              status = "primary",
              collapsible = TRUE,
              p("Box Content")
            )
          )
        )
      )
    )
  )

  server <- function(input, output) {

  }

  shinyApp(ui, server)

}
```

bs4dash_font *bs4Dash fonts parameters*

Description

bs4Dash fonts parameters

Usage

```
bs4dash_font(
  size_base = NULL,
  size_lg = NULL,
  size_sm = NULL,
  size_xs = NULL,
  size_xl = NULL,
  weight_light = NULL,
  weight_normal = NULL,
  weight_bold = NULL,
  family_sans_serif = NULL,
  family_monospace = NULL,
  family_base = NULL
)
```

Arguments

size_base	Base size, this size is used to calculate other size. Must in rem unit.
size_lg	Large size.
size_sm	Small size.
size_xs	Extra small size.
size_xl	Extra large size.
weight_light	Light font weight.
weight_normal	Normal font weight.
weight_bold	Bold font weight.
family_sans_serif	Sans serif font family.
family_monospace	Monospace font family.
family_base	Base font family.

Value

a list that can be used in create_theme.

Examples

```
# Change font size used in bs4Dash
bs4dash_font(
  size_base = "1.5rem",
  weight_bold = 900
)

if (interactive()) {

  library(shiny)
  library(bs4Dash)

  ui <- bs4DashPage(
    title = "bs4Dash Custom Colors",
    navbar = bs4DashNavbar(),
    sidebar = bs4DashSidebar(
      title = "bs4Dash Custom Colors",
      skin = "light",
      bs4SidebarMenu(
        bs4SidebarMenuItem(
          tabName = "tab1",
          text = "UI components"
        )
      )
    ),
    body = bs4DashBody(

      use_theme(create_theme(
        bs4dash_font(
          size_base = "1.5rem",
          weight_bold = 900
        )
      )),

      bs4TabItems(
        bs4TabItem(
          tabName = "tab1",
          tags$div(
            tags$p(
              paste(letters, collapse = "")
            ),
            tags$p(
              style = "font-weight: bold;",
              paste(letters, collapse = "")
            ),
            tags$p(
              style = "font-style: italic;",
              paste(letters, collapse = "")
            )
          ),
          tags$h1("First level title"),
          tags$h2("Second level title"),
          tags$h3("Third level title"),
          tags$h4("Fourth level title"),
          tags$h5("Fifth level title"),
          tags$h6("Sixth level title")
        )
      )
    )
  )

  server <- function(input, output) {

  }

  shinyApp(ui, server)

}
```

`bs4dash_layout` *bs4Dash layout options*

Description

bs4Dash layout options

Usage

```
bs4dash_layout(
  font_size_root = NULL,
  sidebar_width = NULL,
  sidebar_padding_x = NULL,
  sidebar_padding_y = NULL,
  sidebar_mini_width = NULL,
  control_sidebar_width = NULL,
  boxed_layout_max_width = NULL,
  screen_header_collapse = NULL,
  main_bg = NULL,
  content_padding_x = NULL,
  content_padding_y = NULL
)
```

Arguments

`font_size_root` Font size root.

`sidebar_width` Sidebar width.

`sidebar_padding_x`
Sidebar horizontal padding.

`sidebar_padding_y`
Sidebar vertical padding.

`sidebar_mini_width`
Width for mini sidebar.

`control_sidebar_width`
Control sidebar width (the one on the right).

`boxed_layout_max_width`
Max width used in boxed layout.

`screen_header_collapse`
When to show the smaller logo.

`main_bg` Main background color.

`content_padding_x`
Main content horizontal padding.

`content_padding_y`
Main content vertical padding.

Value

a `list` that can be used in `create_theme`.

Examples

```
# Sidebar width
bs4dash_layout(
  sidebar_width = "400px"
)
```

```
if (interactive()) {

  library(shiny)
  library(bs4Dash)

  ui <- bs4DashPage(
    title = "bs4Dash big sidebar",
    navbar = bs4DashNavbar(),
    sidebar = bs4DashSidebar(
      title = "bs4Dash big sidebar",
      skin = "light",
      bs4SidebarMenu(
        bs4SidebarMenuItem(
          tabName = "tab1",
          text = "UI components"
        )
      )
    ),
    body = bs4DashBody(

      use_theme(create_theme(
        bs4dash_layout(
          sidebar_width = "600px"
        )
      )),

      bs4TabItems(
        bs4TabItem(
          tabName = "tab1",
          "Content tab 1"
        )
      )
    )
  )

  server <- function(input, output) {

  }

  shinyApp(ui, server)

}
```

bs4dash_status *bs4Dash status colors*

Description

bs4Dash status colors

Usage

```
bs4dash_status(
  primary = NULL,
  secondary = NULL,
  success = NULL,
  info = NULL,
  warning = NULL,
  danger = NULL,
  light = NULL,
  dark = NULL
)
```

Arguments

`primary`	Default: #0073b7.
`secondary`	Default: #6c757d.
`success`	Default: #28a745.
`info`	Default: #17a2b8.
`warning`	Default: #ffc107.
`danger`	Default: #dc3545.
`light`	Default: #f8f9fa.
`dark`	Default: #343a40.

Value

a `list` that can be used in `create_theme`.

Examples

```
# Change colors used in bs4Dash
bs4dash_status(
  primary = "#F7FE2E",
  secondary = "#01DF3A"
)

if (interactive()) {

  library(shiny)
  library(bs4Dash)

  ui <- bs4DashPage(
    title = "bs4Dash Custom Status",
    # sidebar_collapsed = FALSE,
    navbar = bs4DashNavbar(),
    sidebar = bs4DashSidebar(
      title = "bs4Dash Custom Status",
      skin = "light",
      bs4SidebarMenu(
        bs4SidebarMenuItem(
          tabName = "tab1",
          text = "UI components"
        )
      )
    ),
    body = bs4DashBody(

      use_theme(create_theme(
        bs4dash_status(
          primary = "#F7FE2E",
          secondary = "#01DF3A"
        )
      )),

      bs4TabItems(
        bs4TabItem(
          tabName = "tab1",
          tags$h2("UI components"),
          tags$h4("bs4ValueBox"),
          fluidRow(
            bs4ValueBox(
              value = 150,
              subtitle = "ValueBox with primary status",
              status = "primary",
              icon = "shopping-cart",
              href = "#",
              width = 4
            ),
            bs4ValueBox(
              value = 150,
              subtitle = "ValueBox with secondary status",
              status = "secondary",
              icon = "shopping-cart",
              href = "#",
              width = 4
            )
          ),
          tags$h4("bs4Card"),
          fluidRow(
            bs4Card(
              title = "Card with primary status",
              closable = FALSE,
              width = 6,
              solidHeader = TRUE,
              status = "primary",
              collapsible = TRUE,
              p("Box Content")
            ),
            bs4Card(
              title = "Card with secondary status",
              closable = FALSE,
              width = 6,
              solidHeader = TRUE,
              status = "secondary",
              collapsible = TRUE,
              p("Box Content")
            )
          )
        )
      )
    )
  )

  server <- function(input, output) {

  }

  shinyApp(ui, server)

}
```

bs4Dash_theme *Create a theme for bs4Dash*

Description

Create a theme for bs4Dash

Usage

```
bs4Dash_theme(
  primary = NULL,
  secondary = NULL,
  success = NULL,
  info = NULL,
  warning = NULL,
  danger = NULL,
  ...
)
```

Arguments

primary	A color to be used for hyperlinks, to indicate primary/default actions, and to show active selection state in some Bootstrap components. Generally a bold, saturated color that contrasts with the theme's base colors.
secondary	A color for components and messages that don't need to stand out. (Not supported in Bootstrap 3.)
success	A color for messages that indicate an operation has succeeded. Typically green.
info	A color for messages that are informative but not critical. Typically a shade of blue-green.

warning	A color for warning messages. Typically yellow.
danger	A color for errors. Typically red.
...	Additional AdminLTE variables.

Value

Returns a `sass::sass_bundle()` (list-like) object.

Examples

```
library(shiny)
library(bs4Dash)
library(fresh)

ui <- dashboardPage(
  options = NULL,
  header = dashboardHeader(
    status = "primary",
    title = dashboardBrand(
      title = "My dashboard",
      color = "primary",
      href = "https://adminlte.io/themes/v3",
      image = "https://adminlte.io/themes/v3/dist/img/AdminLTELogo.png"
    )
  ),
  sidebar = dashboardSidebar(
    sidebarMenu(
      menuItem(
        text = "Tab 1",
        tabName = "tab1",
        icon = icon("van-shuttle")
      ),
      menuItem(
        text = "Tab 2",
        tabName = "tab2",
        icon = icon("shuttle-space")
      )
    )
  ),
  body = dashboardBody(
    use_theme(
      bs4Dash_theme(
        primary = "purple",
        success = "yellow",
        danger = "pink",
        "sidebar-light-bg" = "#C38AFF",
        "main-bg" = "#D9BBFF"
      )
    ),
    box(status = "danger", solidHeader = TRUE, title = "Title", "Content"),
    box(status = "primary", solidHeader = TRUE, title = "Title", "Content"),
    box(status = "success", solidHeader = TRUE, title = "Title", "Content")
  ),
  controlbar = dashboardControlbar(),
  title = "DashboardPage"
)


if (interactive()) {
  shinyApp(
    ui = ui,
    server = function(...) {}
  )
}
```

bs4dash_vars *bs4dash custom variables*

Description

Use any AdminLTE or Bootstrap variables to customize a {bs4Dash} theme.

Usage

```
bs4dash_vars(...)
```

Arguments

...	Variables to use, under the form body_bg = "#FFF" or "body-bg" = "#FFF".

Value

a list that can be used in create_theme.

Note

For a full list of available variables, use search_vars_bs4dash.

Examples

```
bs4dash_vars(body_bg = "#FFF")

bs4dash_vars("body-bg" = "#FFF")
```

bs4dash_yiq *bs4Dash color contrast*

Description

These variables allow to customize color used if contrast between a color and its background is under threshold. For example, it's used to choose text color written in bs4ValueBox with background defined by a status.

Usage

```
bs4dash_yiq(contrasted_threshold = NULL, text_dark = NULL, text_light = NULL)
```

Arguments

contrasted_threshold	The yiq lightness value that determines when the lightness of color changes from "dark" to "light". Acceptable values are between 0 and 255.
text_dark	Dark text color.
text_light	Light text color.

Value

a list that can be used in create_theme.

Examples

```
# Contrast colors
bs4dash_yiq(
  contrasted_threshold = 150,
  text_dark = "#007bff", # blue
  text_light = "#dc3545" # red
)

if (interactive()) {

  library(shiny)
  library(bs4Dash)

  ui <- bs4DashPage(
    title = "bs4Dash Contrast",
    navbar = bs4DashNavbar(),
    sidebar = bs4DashSidebar(),
    body = bs4DashBody(

      use_theme(create_theme(
        bs4dash_yiq(
          contrasted_threshold = 180,
          text_dark = "#000",
          text_light = "#dc3545"
        )
      )),

      fluidRow(
        bs4ValueBox(
          value = 120,
          subtitle = "ValueBox with primary status",
          status = "primary",
          icon = "shopping-cart",
          href = "#",
          width = 4
        ),
        bs4ValueBox(
          value = 150,
          subtitle = "ValueBox with danger status",
          status = "danger",
          icon = "shopping-cart",
          href = "#",
          width = 4
        )
      )
    )
  )

  server <- function(input, output) {

  }

  shinyApp(ui, server)

}
```

bs_vars_file *Bootstrap variables from a file*

Description

Bootstrap variables from a file

Usage

```
bs_vars_file(input_file)
```

Arguments

input_file	Path to SCSS file containing variables to use for creating a theme.

Value

a list that can be used in create_theme.

Examples

```
my_vars <- file.path(tempdir(), "custom-vars.scss")
my_theme <- file.path(tempdir(), "theme.css")

# Open template and edit variables
use_vars_template(
  output_file = my_vars,
  theme = "flatly"
)

# Create new theme based on the modified template
create_theme(
  theme = "flatly",
  bs_vars_file(input_file = my_vars),
  output_file = my_theme
)


# Clean up
unlink(my_vars)
unlink(my_theme)
```

---

create_pretty *Create a custom CSS file for pretty-checkbox*

Description

This allow you to change colors of prettyCheckbox, prettyRadioButtons

Usage

```
create_pretty(
  output_file,
  default = NULL,
  primary = NULL,
  success = NULL,
  info = NULL,
  warning = NULL,
  danger = NULL
)
```

Arguments

output_file	Specifies path to output file for compiled CSS.
default	Default color.
primary	Primary color.
success	Success color.

`info`	Info color.
`warning`	Warning color.
`danger`	Danger color.

Value

If `output_file` = `NULL`, the function returns a string value of the compiled CSS. If the output path is specified, the compiled CSS is written to that file and `invisible()` is returned.

Examples

```
# Temporary file
tmp <- file.path(tempdir(), "my-pretty.css")

# Create the new theme
create_pretty(
  output_file = tmp,
  primary = "#FFFF00"
)

# Clean
unlink(tmp)
```

`create_theme` *Create a custom Bootstrap theme*

Description

Allow to customize some CSS variables from Bootstrap themes to be included in Shiny applications.

Usage

```
create_theme(
  ...,
  theme = c("default", "cerulean", "cosmo", "cyborg", "darkly", "flatly", "journal",
    "lumen", "paper", "readable", "sandstone", "simplex", "slate", "spacelab",
    "superhero", "united", "yeti"),
  output_file = NULL,
  include_assets = FALSE
)
```

Arguments

`...`	Lists of CSS variables declared with `bs_vars_*` or `adminlte_*` functions.
`theme`	Base theme to use.
`output_file`	Specifies path to output file for compiled CSS.
`include_assets`	Logical. Only use if `output_file` is not `NULL`, it will copy fonts file used in Bootstrap and Bootswatch themes. Note that output path will be modified to add an intermediate directory `"stylesheets"` where the CSS file will be located.

Value

If `output_file = NULL`, the function returns a string value of the compiled CSS. If the output path is specified, the compiled CSS is written to that file and `invisible()` is returned.

Examples

```
# using a temporary file but use the path you want
tmp <- file.path(tempdir(), "custom-theme.css")

# Create the new theme
create_theme(
  theme = "default",
  bs_vars_color(
    brand_primary = "#75b8d1",
    brand_success = "#c9d175",
    brand_info = "#758bd1",
    brand_warning = "#d1ab75",
    brand_danger = "#d175b8"
  ),
  bs_vars_navbar(
    default_bg = "#75b8d1",
    default_color = "#FFFFFF",
    default_link_color = "#FFFFFF",
    default_link_active_color = "#FFFFFF"
  ),
  output_file = tmp
)

# Use the file created at the path provided
# in your Shiny app by moving it in the
# www/ folder, then use it in UI

library(shiny)
fluidPage(
  theme = "custom-theme.css"
)

# clean up
unlink(tmp)
```

`fresh` *Fresh 'Shiny' Themes*

Description

Customize 'Bootstrap' and 'Bootswatch' themes, like colors, fonts, grid layout, to use in 'Shiny' applications.

Author(s)

Victor Perrier & Fanny Meyer (@dreamRs_fr)

See Also

Useful links:

- `https://github.com/dreamRs/fresh`
- Report bugs at `https://github.com/dreamRs/fresh/issues`

`search_vars` *Search variables in a .scss file*

Description

Search variables in a .scss file

Usage

```
search_vars(file)
```

Arguments

`file`	File path in which to search for variables.

Value

A `data.frame` with 2 columns: `"variable"` and `"value"`.

Examples

```
# Create a scss file with some variables
tmp_scss_file <- tempfile(fileext = ".scss")
writeLines("//Some variables\n $color: red;\n $body-bg: #FFF;", tmp_scss_file)

# Search for variables
search_vars(tmp_scss_file)


# Clean up
unlink(tmp_scss_file)
```

search_vars_adminlte2 *Search AdminLTE 2 shinydashboard variables*

Description

Search AdminLTE 2 shinydashboard variables

Usage

```
search_vars_adminlte2(pattern = NULL)
```

Arguments

`pattern`	A pattern to filter the results.

Value

a `data.frame` with two columns:

- `variable` : name of the variable.
- `value` default value used.

Examples

```
# All AdminLTE2 variables
search_vars_adminlte2()

# Only sidebar related variables
search_vars_adminlte2(pattern = "sidebar")
```

search_vars_bs *Search Bootstrap variables*

Description

Search Bootstrap variables

Usage

```
search_vars_bs(
  pattern = NULL,
  theme = c("default", "cerulean", "cosmo", "cyborg", "darkly", "flatly", "journal",
    "lumen", "paper", "readable", "sandstone", "simplex", "slate", "spacelab",
    "superhero", "united", "yeti")
)
```

Arguments

`pattern`	A pattern to filter the results.
`theme`	Name of the theme for which to search the variables.

Value

a `data.frame` with two columns:

- `variable` : name of the variable.
- `value` default value used.

Examples

```
# List default variables for Bootstrap 3
search_vars_bs()

# Variables for flatly theme
search_vars_bs("flatly")
```

`search_vars_bs4dash` *Search bs4Dash variables*

Description

Search bs4Dash variables

Usage

```
search_vars_bs4dash(pattern = NULL, source = c("adminlte", "bootstrap"))
```

Arguments

`pattern`	A pattern to filter the results.
`source`	Search variables in AdminLTE or Bootstrap or both.

Value

a `data.frame` with three columns:

- `source` : AdminLTE or Bootstrap variable.
- `variable` : name of the variable.
- `value` default value used.

Examples

```
# Retrieve all variables
all_vars <- search_vars_bs4dash()
head(all_vars, 20)

# Search for a pattern
head(search_vars_bs4dash("navbar"))
```

---

use_googlefont *Use online Google font in Shiny application*

Description

Use online Google font in Shiny application

Usage

```
use_googlefont(family)
```

Arguments

family	Name of the family to use, see https://fonts.google.com.

Value

a HTML tag to be included in a UI definition

Examples

```
if (interactive()) {
  library(shiny)
  library(fresh)

  ui <- fluidPage(

    use_googlefont("Saira Stencil One"),
    use_theme(create_theme(
      theme = "default",
      bs_vars_font(
        family_sans_serif = "'Saira Stencil One', cursive"
      )
    )),

    tags$h1("Use a google font (online demo)"),
    fluidRow(
      column(
        width = 6,
        tags$h2("Second level title"),
        tags$h3("Third level title"),
        tags$h4("Fourth level title"),
        tags$h5("Fifth level title"),
        tags$h6("Sixth level title"),
        tags$b("Bold text"),
        tags$p(
          "Lorem ipsum dolor sit amet, consectetur adipiscing elit,",
          " sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.",
          "Ut enim ad minim veniam, quis nostrud exercitation ullamco",
          " laboris nisi ut aliquip ex ea commodo consequat.",
          "Duis aute irure dolor in reprehenderit in voluptate velit",
          " esse cillum dolore eu fugiat nulla pariatur.",
          "Excepteur sint occaecat cupidatat non proident, sunt in",
          " culpa qui officia deserunt mollit anim id est laborum."
        )
      ),
      column(
        width = 6,
        textInput("caption", "Caption", "Data Summary"),
        actionButton("goButton", "Go!"),
        checkboxGroupInput("variable", "Variables to show:",
                           c("Cylinders" = "cyl",
                             "Transmission" = "am",
                             "Gears" = "gear")),
        selectInput("variable", "Variable:",
                    c("Cylinders" = "cyl",
                      "Transmission" = "am",
                      "Gears" = "gear"))
      )
    )
  )

  server <- function(input, output, session) {

  }

  shinyApp(ui, server)
}
```

use_pretty *Use pretty-checkbox CSS in Shiny*

Description

After created new pretty-checkbox CSS with create_pretty, allow to use in Shiny application instead of default shinyWidgets dependency.

Usage

```
use_pretty(path)
```

Arguments

path	Path to the file created with create_pretty, the file must be in www/ directory of the application.

use_theme *Use a CSS theme in Shiny application*

Description

Use a CSS theme in Shiny application

Usage

```
use_theme(theme)
```

Arguments

theme	Either a path to CSS file (if in www/ folder, do not include www/ in path), or a theme generated with create_theme() and argument output_file = NULL or bs4Dash_theme().

Value

HTML tags to be included in a UI definition.

Examples

```
if (interactive()) {
  library(shiny)
  library(fresh)

  ui <- fluidPage(
    use_theme(create_theme(
      theme = "default",
      bs_vars_global(
        body_bg = "#000",
        text_color = "#FFF"
      ),
      bs_vars_wells(
        bg = "#2E2E2E"
      )
    )),
    tags$h1("Inversed color theme"),

    sidebarLayout(
      sidebarPanel(
        "This is the sidebar panel"
      ),
      mainPanel(
        tags$h1("First level title"),
        tags$h2("Second level title"),
        tags$h3("Third level title"),
        tags$h4("Fourth level title"),
        tags$h5("Fifth level title"),
        tags$h6("Sixth level title")
      )
    )
  )

  server <- function(input, output, session) {

  }

  shinyApp(ui, server)
}
```

use_vars_template *Use a template to define SCSS variables*

Description

Open a SCSS template to modify variables, after use bs_vars_file to import those variables and create a theme.

Usage

```
use_vars_template(
  output_file,
 theme = c("default", "cerulean", "cosmo", "cyborg", "darkly", "flatly", "journal",
    "lumen", "paper", "readable", "sandstone", "simplex", "slate", "spacelab",
    "superhero", "united", "yeti"),
  open = interactive()
)
```

Arguments

`output_file`	Path where to create the template, use `".scss"` as file extension.
`theme`	Base theme to use, e.g. `"cosmo"` to start modifying the cosmo theme.
`open`	Open the newly created file for editing? Happens in RStudio, if applicable, or via `utils::file.edit()` otherwise.

Note

After use `bs_vars_file` to use the template.

Examples

```
# For example, we use a temporary file
custom <- tempfile(fileext = ".scss")

# this will open a template
# to modify variables of the flatly theme
use_vars_template(
  output_file = custom,
  theme = "flatly"
)

# after use bs_vars_file() to use the template

# clean up
unlink(custom)
```

Index