Package: bootstrict (via r-universe)

July 2, 2026

Title Strict Bootstrap 5.3 Widgets for Shiny

Version 0.1.0

Description A complete, faithful implementation of the Bootstrap 5.3 component, layout and form library as Shiny UI functions. Every widget mirrors the Bootstrap 5.3 HTML structure one-to-one so that a designer's mockup and SASS variables can be dropped into a Shiny application with minimal deviation from Shiny's own conventions. Interactive components report their state to the server and can be controlled server-side through update functions and proxies. Theming is delegated to 'bslib' (which ships the Bootstrap 5.3 runtime) so designer-provided SASS variable sheets compose natively.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 8.0.0

Depends R (>= 4.1.0)

Imports bslib (>= 0.6.0), htmltools (>= 0.5.4), rlang (>= 1.1.0), shiny (>= 1.7.0)

Suggests golem, testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://github.com/thinkr-open/bootstrict

BugReports https://github.com/thinkr-open/bootstrict/issues

Config/pak/sysreqs cmake make libuv1-dev zlib1g-dev

Repository https://thinkr-open.r-universe.dev

Date/Publication 2026-07-02 12:59:16 UTC

RemoteUrl https://github.com/ThinkR-open/bootstrict

RemoteRef HEAD

RemoteSha 21a59bf2f583eb6f101d21b0b597c28eded5c70a

Contents

bootstrict_dep *The bootstrict HTML dependency (Shiny input bindings + supporting CSS)*

Description

Loads every JavaScript file shipped in `inst/assets/js`, forcing `bootstrict-core.js` first (it defines the `window.bootstrict` namespace that the per-component binding files rely on), followed by the supporting stylesheet. Component binding files are discovered dynamically, so adding a new binding is just a matter of dropping a `.js` file in that directory. The dependency is built once per session and cached (every widget calls this, and the installed files cannot change mid-session).

Usage

```
bootstrict_dep()
```

Value

An htmltools::htmlDependency.

bootstrict_theme *Create a Bootstrap 5 theme for a bootstrict UI*

Description

A thin wrapper around `bslib::bs_theme()` pinned to Bootstrap 5 that also accepts a designer's exported SASS variable sheet. Variables from `variables` are merged with (and overridden by) any variables passed through `...`, then handed to `bslib`.

Usage

```
bootstrict_theme(..., variables = NULL, bootswatch = NULL, preset = NULL)
```

Arguments

`...`	Sass variables / arguments forwarded to `bslib::bs_theme()`. Named values like `primary = "#ff6600"` override Bootstrap defaults.
`variables`	Optional path to a `.scss` variable sheet, or a named list (as returned by `parse_scss_variables()`).
`bootswatch, preset`	Optional Bootswatch / preset name (see `bslib::bs_theme()`).

Value

A `bslib::bs_theme()` object.

Examples

```
if (interactive()) {
  bootstrict_theme(primary = "#ff6600", "font-size-base" = "1rem")
}
```

`bs_accordion` *Bootstrap accordion*

Description

A vertically collapsing set of panels. The value(s) of the currently open panel(s) are reported to the server as `input$id`, and can be driven server-side with `update_bs_accordion()`.

Usage

```
bs_accordion(
  id,
  ...,
  open = NULL,
  multiple = FALSE,
  flush = FALSE,
  class = NULL
)

bs_accordion_panel(
  title,
  ...,
  value = NULL,
  icon = NULL,
  class = NULL,
  body_class = NULL
)
```

Arguments

`id`	Accordion id; open panel value(s) available as `input$id`.
`...`	Panels built with `bs_accordion_panel()`.
`open`	Panel value(s) open initially. Use `TRUE` to open all (only sensible with `multiple = TRUE`), `FALSE`/`NULL` for none.
`multiple`	If `TRUE`, panels stay open independently (Bootstrap "always open"). Otherwise opening one closes the others.
`flush`	If `TRUE`, render edge-to-edge without borders (`.accordion-flush`).
`class`	Extra classes.
`title`	Panel header content.
`value`	Panel identifier reported to the server (defaults to `title`).
`icon`	Optional icon placed before the title.
`body_class`	Extra classes for the panel body.

Value

An accordion tag.

Examples

```
bs_accordion(
  "acc",
  bs_accordion_panel("First", "Panel one body", value = "one"),
  bs_accordion_panel("Second", "Panel two body", value = "two"),
  open = "one"
)
```

---

bs_alert *Bootstrap alert*

Description

Bootstrap alert

Usage

```
bs_alert(..., color = "primary", dismissible = FALSE, class = NULL)

bs_alert_heading(..., level = 4, class = NULL)

bs_alert_link(..., href = "#", class = NULL)
```

Arguments

`...`	Alert content and named HTML attributes.
`color`	Theme colour (`.alert-*`). Defaults to `"primary"`.
`dismissible`	If `TRUE`, add a close button and fade-out behaviour.
`class`	Extra classes.
`level`	Heading level (1-6).
`href`	Link target.

Value

An alert tag.

Examples

```
bs_alert("Well done!", color = "success")
bs_alert("Heads up.", color = "warning", dismissible = TRUE)
```

bs_badge *Bootstrap badge*

Description

A small count / labelling component rendered as a `<span class="badge">`.

Usage

```
bs_badge(..., color = "primary", pill = FALSE, class = NULL)
```

Arguments

`...`	Badge content and named HTML attributes.
`color`	Theme colour (`.text-bg-*`). Defaults to `"primary"`.
`pill`	If `TRUE`, render with fully rounded corners (`.rounded-pill`).
`class`	Extra classes.

Value

A badge tag.

Examples

```
bs_badge("New", color = "success")
```

bs_blockquote *Bootstrap blockquote*

Description

Bootstrap blockquote

Usage

```
bs_blockquote(..., footer = NULL, class = NULL)
```

Arguments

`...`	Quote content and named HTML attributes applied to the `<blockquote>`.
`footer`	Optional source/attribution rendered in a `.blockquote-footer`.
`class`	Extra classes applied to the `<blockquote>`.

Value

A figure tag wrapping the blockquote.

Examples

```
bs_blockquote("A well-known quote.", footer = "Someone famous")
```

`bs_breadcrumb` *Bootstrap breadcrumb*

Description

A navigation hierarchy. Compose with `bs_breadcrumb_item()`.

Usage

```
bs_breadcrumb(..., divider = NULL, label = "breadcrumb", class = NULL)

bs_breadcrumb_item(..., active = FALSE, href = NULL, class = NULL)
```

Arguments

`...`	Breadcrumb items built with `bs_breadcrumb_item()`, plus named HTML attributes applied to the <nav>.
`divider`	Optional custom divider character (e.g. ">"). When a string, it is set via the `--bs-breadcrumb-divider` CSS variable on the <nav>.
`label`	Accessible label for the <nav> (`aria-label`).
`class`	Extra classes applied to the <nav>.
`active`	If `TRUE`, mark the item as the current page (no link).
`href`	Optional link target. Ignored when `active = TRUE`.

Value

A breadcrumb <nav> tag.

Examples

```
bs_breadcrumb(
  bs_breadcrumb_item("Home", href = "#"),
  bs_breadcrumb_item("Library", active = TRUE)
)
```

bs_button *Bootstrap button*

Description

Renders a Bootstrap 5 button. When `id` is supplied the button is wired as a Shiny action button: `input$id` holds the click count, exactly like `shiny::actionButton()`.

Usage

```
bs_button(
  id = NULL,
  label = NULL,
  ...,
  color = "primary",
  outline = FALSE,
  size = NULL,
  disabled = FALSE,
  href = NULL,
  type = "button",
  class = NULL
)
```

Arguments

`id`	Optional input id. When set, the click count is reported as `input$id`.
`label`	Button label (text or tags). Can also be passed via `...`.
`...`	Additional content and named HTML attributes.
`color`	One of the eight Bootstrap theme colours, or `"link"`.
`outline`	If `TRUE`, an outline button (`.btn-outline-*`).
`size`	`"sm"` or `"lg"`.
`disabled`	If `TRUE`, the button is disabled.
`href`	Optional URL; renders an `<a>` styled as a button.
`type`	Button `type` attribute (`"button"`, `"submit"`, `"reset"`).
`class`	Extra classes.

Value

A button (or anchor) tag.

Examples

```
bs_button("go", "Go", color = "primary")
bs_button(label = "Cancel", color = "secondary", outline = TRUE)
```

bs_button_group *Bootstrap button group / toolbar*

Description

Bootstrap button group / toolbar

Usage

```
bs_button_group(..., size = NULL, vertical = FALSE, label = NULL, class = NULL)

bs_button_toolbar(..., label = NULL, class = NULL)
```

Arguments

...	Buttons (bs_button()) and named HTML attributes.
size	"sm" or "lg".
vertical	If TRUE, stack vertically (.btn-group-vertical).
label	Accessible label (aria-label).
class	Extra classes.

Value

A button group / toolbar tag.

Examples

```
bs_button_group(bs_button(label = "Left"), bs_button(label = "Right"))
```

bs_card *Bootstrap card*

Description

A flexible content container. Compose with bs_card_header(), bs_card_body(), bs_card_footer(), bs_card_img() and the card text helpers.

Usage

```
bs_card(..., color = NULL, border = NULL, class = NULL)

bs_card_body(..., class = NULL)

bs_card_header(..., class = NULL)

bs_card_footer(..., class = NULL)

bs_card_title(..., level = 5, class = NULL)

bs_card_subtitle(..., level = 6, class = NULL)

bs_card_text(..., class = NULL)

bs_card_link(..., href = "#", class = NULL)

bs_card_img(
  src,
  position = c("top", "bottom", "overlay"),
  alt = NULL,
  ...,
  class = NULL
)

bs_card_img_overlay(..., class = NULL)

bs_card_group(..., class = NULL)
```

Arguments

`...`	Card content (headers, bodies, images, ...) and named HTML attributes.
`color`	Background theme colour (`.text-bg-*`), one of the Bootstrap theme colours.
`border`	Border theme colour (`.border-*`).
`class`	Extra classes.
`level`	Heading level (1-6) for the card title / subtitle.
`href`	Link target for `bs_card_link()`.
`src`	Image source URL.
`position`	Image placement: `"top"`, `"bottom"`, or `"overlay"` (use together with `bs_card_img_overlay()`).
`alt`	Image alt text.

Value

A card tag.

Examples

```
bs_card(
  bs_card_header("Featured"),
  bs_card_body(
    bs_card_title("Card title"),
    bs_card_text("Some quick example text.")
  )
)
```

`bs_carousel` *Bootstrap carousel*

Description

A slideshow component for cycling through a series of items (`bs_carousel_item()`s). The 0-based index of the active slide is reported to the server as `input$id`, and can be driven server-side with `update_bs_carousel()`.

Usage

```
bs_carousel(
  id,
  ...,
  indicators = TRUE,
  controls = TRUE,
  fade = FALSE,
  autoplay = TRUE,
  interval = NULL,
  dark = FALSE,
  class = NULL
)

bs_carousel_item(
  ...,
  active = FALSE,
  interval = NULL,
  caption = NULL,
  class = NULL
)
```

Arguments

`id`	Carousel id; the active slide index is available as `input$id`.
`...`	Items built with `bs_carousel_item()` (and named HTML attributes).
`indicators`	If `TRUE`, render the clickable slide indicators.
`controls`	If `TRUE`, render the previous / next control buttons.

fade	If TRUE, crossfade between slides instead of sliding (.carousel-fade).
autoplay	If TRUE (default), start cycling on load (data-bs-ride="carousel"). When FALSE, the attribute is omitted so the carousel only ever advances on user interaction (Bootstrap's data-bs-ride="true" would resume autoplay after the first interaction).
interval	Cycling interval in milliseconds (sets data-bs-interval).
dark	If TRUE, the dark variant via data-bs-theme="dark" (the Bootstrap 5.3 idiom; .carousel-dark is deprecated).
class	Extra classes.
active	If TRUE, this item is shown first. Exactly one item per carousel is active; bs_carousel() defaults the first item if none is set.
caption	Optional caption content placed in a .carousel-caption (hidden on small screens, d-none d-md-block).

Value

A carousel tag.

Examples

```
bs_carousel(
  "demo",
  bs_carousel_item(htmltools::img(src = "1.jpg"), active = TRUE),
  bs_carousel_item(htmltools::img(src = "2.jpg"))
)
```

---

bs_checkbox_group_input

Bootstrap checkbox group

Description

Delegates to shiny::checkboxGroupInput() and rewrites the markup to Bootstrap 5 .form-check shape so the value stays available as input$id.

Usage

```
bs_checkbox_group_input(
  id,
  label = NULL,
  choices,
  selected = NULL,
  ...,
  inline = FALSE,
  reverse = FALSE,
  help = NULL,
  width = NULL
)
```

Arguments

id	Input id; selected value available as input$id.
label	Input label.
choices	Named or unnamed vector / list of choices.
selected	Initially selected value(s).
...	Extra attributes applied to each radio <input>.
inline	If TRUE, lay the choices out horizontally.
reverse	If TRUE, put the label before the control (.form-check-reverse, Bootstrap 5.2).
help	Help text rendered below the control (.form-text).
width	CSS width (e.g. "100%", "200px").

Value

A form control tag.

See Also

shiny::checkboxGroupInput()

Examples

```
bs_checkbox_group_input("opts", "Options", c("A", "B", "C"), selected = "A")
```

---

bs_checkbox_input *Bootstrap checkbox input*

Description

Bootstrap checkbox input

Usage

```
bs_checkbox_input(
  id,
  label = NULL,
  value = FALSE,
  ...,
  switch = FALSE,
  reverse = FALSE,
  help = NULL,
  width = NULL
)

bs_switch_input(
  id,
```

```
  label = NULL,
  value = FALSE,
  ...,
  reverse = FALSE,
  help = NULL,
  width = NULL
)
```

Arguments

id	Input id; value available as input$id.
label	Input label.
value	Initial checked state.
...	Extra attributes applied to the <input> element.
switch	If TRUE, render as a toggle switch (.form-switch).
reverse	If TRUE, put the label before the control (.form-check-reverse, Bootstrap 5.2).
help	Help text rendered below the control (.form-text).
width	CSS width (e.g. "100%", "200px").

Value

A form control tag.

See Also

shiny::checkboxInput()

Examples

```
bs_checkbox_input("agree", "I agree", TRUE)
bs_checkbox_input("dark", "Dark mode", switch = TRUE)
```

---

bs_close_button *Bootstrap close button*

Description

Bootstrap close button

Usage

```
bs_close_button(..., label = "Close", white = FALSE, class = NULL)
```

Arguments

...	Named HTML attributes (e.g. data-bs-dismiss).
label	Accessible label.
white	If TRUE, the variant for dark backgrounds (data-bs-theme="dark" on the button — the Bootstrap 5.3 idiom; .btn-close-white is deprecated).
class	Extra classes.

Value

A button tag.

---

bs_col *Bootstrap grid column*

Description

Bootstrap grid column

Usage

```
bs_col(
  ...,
  width = NULL,
  sm = NULL,
  md = NULL,
  lg = NULL,
  xl = NULL,
  xxl = NULL,
  offset = NULL,
  order = NULL,
  align_self = NULL,
  class = NULL
)
```

Arguments

...	Content, and named HTML attributes.
width	Base column span: an integer 1-12, "auto", or TRUE for an equal-width column. NULL (default) yields a bare .col.
sm, md, lg, xl, xxl	Per-breakpoint spans (integer, "auto" or TRUE).
offset	Column offset. A single value applies at the base breakpoint; a named list sets per-breakpoint offsets (e.g. list(md = 2)).
order	Column order (order-*): integer 0-5, "first" or "last". A named list sets per-breakpoint order.
align_self	Vertical self-alignment: "start", "center", "end".
class	Extra classes.

Value

A column tag.

Examples

```
bs_col(width = 6, md = 4, "content")
```

`bs_collapse` *Bootstrap collapse*

Description

A toggleable container that shows or hides content. Pair it with a `bs_collapse_trigger()` (declarative, no server round trip) and/or drive it from the server with `update_bs_collapse()`. Its shown/hidden state is reported to the server as `input$id` (`TRUE` when visible).

Usage

```
bs_collapse(id, ..., open = FALSE, horizontal = FALSE, class = NULL)
```

Arguments

`id`	Collapse id; visibility state available as `input$id`.
`...`	Collapse content and named HTML attributes.
`open`	If `TRUE`, the collapse is visible initially (`.show`).
`horizontal`	If `TRUE`, collapse transitions width instead of height (`.collapse-horizontal`).
`class`	Extra classes.

Value

A collapse tag.

Examples

```
bs_collapse("more", "Hidden content revealed on toggle.")
```

bs_collapse_trigger *Trigger a collapse from the UI*

Description

Renders a control that toggles the bs_collapse() whose id matches target, using Bootstrap's declarative data-bs-toggle="collapse" attributes (no server round trip required).

Usage

```
bs_collapse_trigger(target, ..., button = TRUE, expanded = FALSE, class = NULL)
```

Arguments

target	Id of the bs_collapse() to toggle.
...	Content (label) and named HTML attributes.
button	If TRUE (default), render a <button class="btn">; otherwise render an <a role="button">.
expanded	Initial expanded state of the trigger. Set to TRUE when the target is a bs_collapse(open = TRUE) so the initial aria-expanded / .collapsed state is correct.
class	Extra classes.

Value

A button (or anchor) tag.

Examples

```
bs_collapse_trigger("more", "Toggle")
```

bs_color_input *Bootstrap colour input*

Description

A native <input type="color" class="form-control form-control-color"> whose value (a "#rrggbb" string) is reported as input$id. Drive it server-side with update_bs_color().

Usage

```
bs_color_input(
  id,
  label = NULL,
  value = "#000000",
  ...,
  help = NULL,
  width = NULL
)
```

Arguments

id	Input id; value available as input$id.
label	Input label.
value	Initial colour as a hex string (e.g. "#0d6efd").
...	Extra attributes applied to the <input> element.
help	Help text rendered below the control (.form-text).
width	CSS width (e.g. "100%", "200px").

Value

A form control tag.

Examples

```
bs_color_input("col", "Pick a colour", value = "#0d6efd")
```

---

bs_container *Bootstrap container*

Description

Bootstrap container

Usage

```
bs_container(..., fluid = FALSE, breakpoint = NULL, class = NULL)
```

Arguments

...	Content, and named HTML attributes.
fluid	If TRUE, a full-width .container-fluid. Ignored when breakpoint is set.
breakpoint	One of "sm", "md", "lg", "xl", "xxl" for a responsive .container-{breakpoint}.
class	Extra classes.

Value

A container tag.

See Also

bs_row(), bs_col()

Examples

```
bs_container(bs_row(bs_col("a"), bs_col("b")))
```

`bs_date_input` *Bootstrap date input*

Description

Delegates to `shiny::dateInput()` and adds the Bootstrap `.form-label` / `.form-control` affordances.

Usage

```
bs_date_input(
  id,
  label = NULL,
  value = NULL,
  ...,
  min = NULL,
  max = NULL,
  help = NULL,
  width = NULL
)
```

Arguments

`id`	Input id; selected value available as `input$id`.
`label`	Input label.
`value`	Initial date (a `Date` or `"yyyy-mm-dd"` string).
`...`	Extra attributes applied to each radio `<input>`.
`min, max`	Minimum / maximum selectable date.
`help`	Help text rendered below the control (`.form-text`).
`width`	CSS width (e.g. `"100%"`, `"200px"`).

Value

A form control tag.

See Also

`shiny::dateInput()`

Examples

```
bs_date_input("day", "Pick a day", value = "2026-06-26")
```

`bs_date_range_input` *Bootstrap date range input*

Description

Delegates to `shiny::dateRangeInput()` and adds the Bootstrap `.form-label` / `.form-control` affordances.

Usage

```
bs_date_range_input(
  id,
  label = NULL,
  start = NULL,
  end = NULL,
  ...,
  min = NULL,
  max = NULL,
  help = NULL,
  width = NULL
)
```

Arguments

`id`	Input id; selected value available as `input$id`.
`label`	Input label.
`start`, `end`	Initial start / end dates.
`...`	Extra attributes applied to each radio `<input>`.
`min`, `max`	Minimum / maximum selectable date.
`help`	Help text rendered below the control (`.form-text`).
`width`	CSS width (e.g. `"100%"`, `"200px"`).

Value

A form control tag.

See Also

`shiny::dateRangeInput()`

Examples

```
bs_date_range_input("range", "Period", start = "2026-01-01", end = "2026-12-31")
```

bs_display_heading *Bootstrap display heading*

Description

A larger, slightly more opinionated heading style (`.display-*`).

Usage

```
bs_display_heading(..., level = 1, class = NULL)
```

Arguments

`...`	Heading content and named HTML attributes.
`level`	Display size / heading level (1-6).
`class`	Extra classes.

Value

A heading tag.

Examples

```
bs_display_heading("Display heading", level = 2)
```

bs_dropdown *Bootstrap dropdown*

Description

A toggleable menu of links, headers and dividers. Compose the menu with `bs_dropdown_item()`, `bs_dropdown_divider()`, `bs_dropdown_header()` and `bs_dropdown_text()`. Bootstrap drives the toggle; give an item an `id` to wire it as a Shiny action button (`input$id`).

Usage

```
bs_dropdown(
  label,
  ...,
  color = "secondary",
  outline = FALSE,
  size = NULL,
  split = FALSE,
  direction = "down",
  dark = FALSE,
```

```
  align = NULL,
  class = NULL
)

bs_dropdown_item(
  ...,
  id = NULL,
  href = "#",
  active = FALSE,
  disabled = FALSE,
  class = NULL
)

bs_dropdown_divider(class = NULL)

bs_dropdown_header(..., level = 6, class = NULL)

bs_dropdown_text(..., class = NULL)
```

Arguments

label	Toggle button label (text or tags).
...	Menu contents (dropdown items, dividers, headers, text) and named HTML attributes (forwarded to the wrapper).
color	Toggle button theme colour, one of the Bootstrap theme colours or "link".
outline	If TRUE, an outline toggle button (.btn-outline-*).
size	Toggle button size: "sm" or "lg".
split	If TRUE, render a split button (a normal action button plus a separate toggle caret).
direction	Menu drop direction: "down", "up", "end" or "start".
dark	If TRUE, a dark dropdown via data-bs-theme="dark" on the wrapper (the Bootstrap 5.3 idiom; .dropdown-menu-dark is deprecated).
align	Menu alignment. "end" right-aligns the menu (.dropdown-menu-end); a named list such as list(lg = "end") produces a responsive alignment (.dropdown-menu-lg-end).
class	Extra classes for the wrapper.
id	Optional input id. When set, the item becomes a Shiny action button and its click count is reported as input$id.
href	Link target.
active	If TRUE, mark the item active (.active).
disabled	If TRUE, mark the item disabled (.disabled).
level	Heading level (1-6) for the dropdown header.

Value

A dropdown tag.

Examples

```
bs_dropdown("Menu", bs_dropdown_item("Action", id = "act"))
```

`bs_figure` *Bootstrap figure*

Description

Group an image with a caption. Compose `bs_figure()` with `bs_figure_img()` and `bs_figure_caption()`.

Usage

```
bs_figure(..., class = NULL)

bs_figure_img(src, ..., alt = NULL, class = NULL)

bs_figure_caption(..., class = NULL)
```

Arguments

`...`	Figure content (image, caption) and named HTML attributes.
`class`	Extra classes.
`src`	Image source URL.
`alt`	Alternative text.

Value

A figure tag.

Examples

```
bs_figure(
  bs_figure_img("photo.jpg", alt = "A photo"),
  bs_figure_caption("A caption for the image.")
)
```

bs_file_input *Bootstrap file input*

Description

Delegates to `shiny::fileInput()` and adapts its markup to Bootstrap 5.

Usage

```
bs_file_input(
  id,
  label = NULL,
  ...,
  multiple = FALSE,
  accept = NULL,
  button_label = "Browse...",
  placeholder = "No file selected",
  width = NULL
)
```

Arguments

`id`	Input id; selected value available as `input$id`.
`label`	Input label.
`...`	Extra attributes applied to the `<input type="file">` element itself (e.g. `capture`, `webkitdirectory`).
`multiple`	Allow selecting more than one file.
`accept`	Character vector of accepted MIME types / extensions.
`button_label`	Label shown on the browse button.
`placeholder`	Placeholder text shown before a file is chosen.
`width`	CSS width (e.g. `"100%"`, `"200px"`).

Details

shiny hides the real `<input type="file">` off-screen (`top: -99999px`) and relies on Bootstrap 3 `.btn-file` CSS to bring it back over the browse button. That CSS is absent under Bootstrap 5, so clicking the button focuses the off-screen input and the browser scrolls the page to the top. We fix this by overlaying the input on the button (invisible, `opacity: 0`) so a click lands on it directly, and restyle the browse button as a real Bootstrap 5 button.

Value

A form control tag.

See Also

shiny::fileInput()

Examples

```
bs_file_input("upload", "Upload a file", accept = ".csv")
```

---

bs_floating_label *Bootstrap floating label*

Description

Reshape a control built by one of the bs_*_input() constructors (e.g. bs_text_input(), bs_select_input()) into a Bootstrap 5 floating-label group, where the label animates into the control's border. The existing control and its Shiny input wiring are preserved (the element is moved, not rebuilt), so the value still reports as input$id.

Usage

```
bs_floating_label(input, label = NULL, class = NULL)
```

Arguments

input	A control tag tree produced by a bs_*_input() constructor.
label	Floating label text. If NULL, the control's existing label text is reused.
class	Extra classes.

Value

A .form-floating wrapper tag.

Examples

```
bs_floating_label(bs_text_input("email", "Email address"))
```

bs_form *Bootstrap form element*

Description

A plain <form> container for grouping form controls.

Usage

```
bs_form(..., novalidate = FALSE, class = NULL)
```

Arguments

...	Form content (controls, layout, buttons) and named HTML attributes.
novalidate	If TRUE, add the novalidate attribute to disable the browser's native validation UI (useful with custom validation feedback).
class	Extra classes.

Value

A form tag.

Examples

```
bs_form(
  bs_text_input("email", "Email"),
  bs_button("Submit", color = "primary")
)
```

bs_form_label *Bootstrap form label*

Description

A <label class="form-label"> tied to a control via its for attribute.

Usage

```
bs_form_label(for_id, ..., class = NULL)
```

Arguments

for_id	The id of the control this label describes (rendered as the for attribute).
...	Label content and named HTML attributes.
class	Extra classes.

Value

A label tag.

Examples

```
bs_form_label("email", "Email address")
```

`bs_form_text` *Bootstrap form help text*

Description

Muted helper text rendered below a control (`.form-text`).

Usage

```
bs_form_text(..., class = NULL)
```

Arguments

`...`	Help content and named HTML attributes.
`class`	Extra classes.

Value

A div tag.

Examples

```
bs_form_text("Must be 8-20 characters long.")
```

`bs_hstack` *Bootstrap stacks (horizontal / vertical flex layouts)*

Description

Shorthand flex helpers added in Bootstrap 5.1. `bs_hstack()` lays children out in a row, `bs_vstack()` in a column.

Usage

```
bs_hstack(..., gap = NULL, class = NULL)

bs_vstack(..., gap = NULL, class = NULL)
```

Arguments

...	Content, and named HTML attributes.
gap	Spacing between items (gap-*), an integer 0-5.
class	Extra classes.

Value

A stack tag.

Examples

```
bs_hstack(bs_button(label = "A"), bs_button(label = "B"), gap = 2)
bs_vstack(bs_alert("one"), bs_alert("two"), gap = 3)
```

---

bs_icon_link *Bootstrap icon link*

Description

The Bootstrap 5.3 icon-link helper: an inline flex <a> that pairs an icon (e.g. an SVG) with text, with an optional hover translation of the icon (.icon-link-hover).

Usage

```
bs_icon_link(..., href = "#", hover = FALSE, class = NULL)
```

Arguments

...	Link content (icon and text) and named HTML attributes.
href	Link target.
hover	If TRUE, translate the icon on hover (.icon-link-hover).
class	Extra classes.

Value

An anchor tag.

Examples

```
bs_icon_link("Icon link", href = "#")
```

bs_img *Bootstrap image*

Description

Bootstrap image

Usage

```
bs_img(
  src,
  ...,
  fluid = FALSE,
  thumbnail = FALSE,
  rounded = FALSE,
  object_fit = NULL,
  alt = NULL,
  class = NULL
)
```

Arguments

src	Image source URL.
...	Extra named HTML attributes applied to the <img>.
fluid	Make the image responsive (.img-fluid).
thumbnail	Render with a rounded thumbnail border (.img-thumbnail).
rounded	Add rounded corners (.rounded).
object_fit	How the image fills its box (Bootstrap 5.3 .object-fit-* utility): "contain", "cover", "fill", "scale" (scale-down) or "none".
alt	Alternative text.
class	Extra classes.

Value

An image tag.

Examples

```
bs_img("logo.png", alt = "Logo", fluid = TRUE)
```

bs_input_group *Bootstrap input group*

Description

Groups one or more form controls together with add-ons (text, buttons, dropdowns) on a single line.

Usage

```
bs_input_group(..., size = NULL, class = NULL)

bs_input_group_text(..., class = NULL)
```

Arguments

...	Controls and add-ons (e.g. bs_input_group_text(), bs_text_input(), bs_button()) and named HTML attributes.
size	Control size: "sm" or "lg".
class	Extra classes.

Details

When a full bs_*_input() is passed, only its bare control is kept so it sits flush with the add-ons: the input's label and help text are dropped (compose them outside the group). Inputs whose Shiny binding lives on their container — bs_date_input(), bs_date_range_input(), bs_file_input(), bs_radio_input(), bs_checkbox_group_input() — cannot be unwrapped this way and raise an error.

Value

An input-group tag.

Examples

```
bs_input_group(
  bs_input_group_text("@"),
  bs_text_input("user", placeholder = "Username")
)
```

bs_lead *Bootstrap lead paragraph*

Description

A standout opening paragraph (`.lead`).

Usage

```
bs_lead(..., class = NULL)
```

Arguments

`...`	Paragraph content and named HTML attributes.
`class`	Extra classes.

Value

A paragraph tag.

Examples

```
bs_lead("This is a lead paragraph.")
```

bs_list_group *Bootstrap list group*

Description

A flexible container for displaying a series of content. Compose with `bs_list_group_item()`. When `id` is supplied the group becomes selectable: clicking an action item (`action = TRUE` or `href`) reports that item's `value` (its `data-value`) as `input$id`, and the selection can be driven server-side with `update_bs_list_group()`.

Usage

```
bs_list_group(
  id = NULL,
  ...,
  flush = FALSE,
  numbered = FALSE,
  horizontal = FALSE,
  class = NULL
)

bs_list_group_item(
```

```
  ...,
  value = NULL,
  active = FALSE,
  disabled = FALSE,
  color = NULL,
  action = FALSE,
  href = NULL,
  class = NULL
)
```

Arguments

id	Optional list group id. When set, the active item's value is reported as `input$id` and the group is wired for selection.
...	List items built with `bs_list_group_item()`, plus named HTML attributes.
flush	If `TRUE`, render edge-to-edge without an outer border and rounded corners (`.list-group-flush`).
numbered	If `TRUE`, render a numbered list (`<ol class="list-group list-group-numbered">`).
horizontal	Lay items out horizontally. `TRUE` for `.list-group-horizontal`, or a breakpoint string (`"sm"`, `"md"`, `"lg"`, `"xl"`, `"xxl"`) for `.list-group-horizontal-{bp}`.
class	Extra classes.
value	Item value reported to the server when selected (its `data-value`). Only meaningful in a selectable group (when the parent `bs_list_group()` has an `id`).
active	If `TRUE`, mark the item as the active/selected one.
disabled	If `TRUE`, mark the item disabled.
color	Contextual theme colour (`.list-group-item-*`).
action	If `TRUE`, render an actionable `<button>` item (`.list-group-item-action`). Ignored when `href` is supplied (which always renders an actionable `<a>`).
href	Link target; renders the item as an `<a>` (always actionable).

Value

A list group tag.

Examples

```
bs_list_group(
  bs_list_group_item("An item"),
  bs_list_group_item("A second item", active = TRUE)
)
bs_list_group_item("A link item", href = "#", value = "a")
```

bs_list_unstyled *Bootstrap unstyled / inline lists*

Description

Remove a list's default styling (`.list-unstyled`) or lay items out inline (`.list-inline`). Each unnamed `...` argument becomes one `<li>`; named arguments become attributes of the `<ul>`.

Usage

```
bs_list_unstyled(..., class = NULL)

bs_list_inline(..., class = NULL)
```

Arguments

`...`	List items (unnamed) and named HTML attributes for the `<ul>`.
`class`	Extra classes.

Value

An unordered list tag.

Examples

```
bs_list_unstyled("First", "Second", "Third")
bs_list_inline("One", "Two", "Three")
```

bs_modal *Bootstrap modal*

Description

A faithful Bootstrap 5 modal dialog. The modal's open/closed state is reported to the server as `input$id` (`TRUE` when shown), and can be driven server-side with `show_bs_modal()`, `hide_bs_modal()` and `toggle_bs_modal()`.

Usage

```
bs_modal(
  id,
  ...,
  title = NULL,
  footer = NULL,
  size = NULL,
  centered = FALSE,
```

```
  scrollable = FALSE,
  fullscreen = FALSE,
  backdrop = TRUE,
  keyboard = TRUE,
  class = NULL
)

bs_modal_header(..., class = NULL)

bs_modal_title(..., level = 1, class = NULL)

bs_modal_body(..., class = NULL)

bs_modal_footer(..., class = NULL)
```

Arguments

id	Modal id; open state available as input$id.
...	Modal body content and named HTML attributes applied to the root.
title	Optional header title. When supplied, a .modal-header with a title and a close button is rendered.
footer	Optional footer content (e.g. buttons), placed in a .modal-footer.
size	Dialog size: "sm", "lg" or "xl".
centered	If TRUE, vertically centre the dialog (.modal-dialog-centered).
scrollable	If TRUE, scroll long bodies inside the dialog (.modal-dialog-scrollable).
fullscreen	TRUE for an always-fullscreen modal, or a breakpoint ("sm"/"md"/"lg"/"xl"/"xxl") for fullscreen below that breakpoint (.modal-fullscreen-{bp}-down).
backdrop	Backdrop behaviour: TRUE (backdrop shown, dismiss on outside click), "static" (backdrop shown, do not dismiss on outside click) or FALSE (no backdrop at all).
keyboard	If FALSE, the modal cannot be closed with the Escape key.
class	Extra classes.
level	Heading level (1-6) for the modal title.

Details

For full control over the dialog structure (in place of the title and footer shortcuts) compose the body yourself with bs_modal_header(), bs_modal_body() and bs_modal_footer().

Value

A modal tag.

Examples

```
bs_modal("info", "Modal body text.", title = "Heads up")
```

bs_modal_trigger *Trigger a modal from the UI*

Description

Renders a button that opens the modal whose id matches `target`, using Bootstrap's declarative `data-bs-toggle="modal"` attributes (no server round trip required).

Usage

```
bs_modal_trigger(target, ..., class = NULL)
```

Arguments

`target`	Id of the `bs_modal()` to open.
`...`	Content (label) and named HTML attributes, forwarded to `bs_button()`.
`class`	Extra classes.

Value

A button tag.

Examples

```
bs_modal_trigger("info", "Open modal", color = "primary")
```

bs_nav *Bootstrap navigation list*

Description

A static navigation container rendered as a Bootstrap `<ul class="nav">`. Compose it with `bs_nav_item()` and `bs_nav_link()`. For an interactive, server-reporting tabset use `bs_tabset()` instead.

Usage

```
bs_nav(
  ...,
  type = NULL,
  fill = FALSE,
  justified = FALSE,
  vertical = FALSE,
  class = NULL
)
```

```
bs_nav_item(..., class = NULL)

bs_nav_link(
  ...,
  href = "#",
  active = FALSE,
  disabled = FALSE,
  id = NULL,
  class = NULL
)
```

Arguments

...	Navigation items (bs_nav_item() / bs_nav_link()) and named HTML attributes.
type	Visual style: "tabs", "pills" or "underline" (Bootstrap 5.3) — default NULL for a plain nav).
fill	If TRUE, items expand to fill available width (.nav-fill).
justified	If TRUE, items get equal width (.nav-justified).
vertical	If TRUE, stack items vertically (.flex-column).
class	Extra classes.
href	Link target.
active	If TRUE, mark the link as the active page (adds .active and aria-current="page").
disabled	If TRUE, mark the link disabled (.disabled).
id	Optional element id.

Value

A nav tag.

Examples

```
bs_nav(
  bs_nav_item(bs_nav_link("Active", active = TRUE)),
  bs_nav_item(bs_nav_link("Link")),
  type = "tabs"
)
```

---

bs_navbar *Bootstrap navbar*

Description

A responsive navigation header. Compose with bs_navbar_brand(), bs_navbar_nav() (containing bs_nav_item() / bs_nav_link() from the nav-tabs group), bs_navbar_text() and, optionally, bs_dropdown(). The navbar collapses behind a toggler below the expand breakpoint.

Usage

```
bs_navbar(
  ...,
  brand = NULL,
  id = NULL,
  expand = "lg",
  bg = NULL,
  theme = NULL,
  placement = NULL,
  fluid = TRUE,
  class = NULL
)

bs_navbar_brand(..., href = "#", class = NULL)

bs_navbar_nav(..., scroll = FALSE, scroll_height = NULL, class = NULL)

bs_navbar_text(..., class = NULL)
```

Arguments

`...`	Navbar content (brand, nav lists, text, ...) placed inside the collapsible container, plus named HTML attributes for the <nav>.
`brand`	Optional brand element, typically `bs_navbar_brand()`, rendered before the toggler so it stays visible when collapsed.
`id`	Navbar id; used to wire the toggler to its collapsible region (`"<id>-collapse"`). Defaults to a unique auto-generated id so multiple navbars on a page never collide.
`expand`	Breakpoint at which the navbar expands: one of `"sm"`, `"md"`, `"lg"`, `"xl"`, `"xxl"`, or `TRUE` to always expand (`.navbar-expand`). Use `FALSE`/`NULL` to never expand (always collapsed).
`bg`	Background colour (`.bg-*`): one of the Bootstrap theme colours, or `"body"`, `"body-secondary"`, `"body-tertiary"` (the Bootstrap 5.3 navbar default), `"white"`, `"black"`, `"transparent"`.
`theme`	Colour scheme applied via `data-bs-theme` (the Bootstrap 5.3 colour-modes idiom; `.navbar-light`/`.navbar-dark` are deprecated): `"light"` or `"dark"`.
`placement`	Fixed/sticky placement: one of `"fixed-top"`, `"fixed-bottom"`, `"sticky-top"`, `"sticky-bottom"`.
`fluid`	If `TRUE` (default) use a full-width `.container-fluid`, otherwise a fixed-width `.container`.
`class`	Extra classes for the <nav>.
`href`	Link target for the brand.
`scroll`	If `TRUE`, enable vertical scrolling within the collapsed navbar nav (`.navbar-nav-scroll`, Bootstrap 5.1).
`scroll_height`	Max scroll height (sets `--bs-scroll-height`), e.g. `"75vh"` or `"200px"`. Only applies when `scroll = TRUE`.

Value

A navbar tag.

Examples

```
bs_navbar(brand = bs_navbar_brand("Navbar"), bg = "light", theme = "light")
bs_navbar_brand("Acme", href = "/")
bs_navbar_nav(bs_nav_item(bs_nav_link("Home", active = TRUE)))
bs_navbar_text("Signed in as Mark Otto")
```

---

bs_notify_toast *Pop a transient toast notification from the server*

Description

Builds a toast on the client and shows it, much like `shiny::showNotification()`. A toast container is created on demand at `placement` if one is not already present, and the toast removes itself from the DOM once hidden.

Usage

```
bs_notify_toast(
  body,
  ...,
  title = NULL,
  color = NULL,
  autohide = TRUE,
  delay = 5000,
  placement = "top-end",
  session = shiny::getDefaultReactiveDomain()
)
```

Arguments

`body`	Notification body text. Plain text only (it is inserted with `textContent` client-side, so markup is not interpreted); tags raise an error.
`...`	Reserved for future extensions; must be empty.
`title`	Optional header title. Plain text only, like `body`.
`color`	Optional theme colour applied as a `.text-bg-*` background.
`autohide`	If `TRUE` (default), hide the toast automatically after `delay` milliseconds. Use `FALSE` for a persistent notification the user must dismiss.
`delay`	Delay in milliseconds before the toast auto-hides.
`placement`	Container placement (see `bs_toast_container()`).
`session`	The Shiny session.

Value

Invisibly `NULL`, called for its side effect.

Examples

```
if (interactive()) bs_notify_toast("Saved!", title = "Status", color = "success")
```

`bs_numeric_input` *Bootstrap numeric input*

Description

Bootstrap numeric input

Usage

```
bs_numeric_input(
  id,
  label = NULL,
  value = NULL,
  ...,
  min = NULL,
  max = NULL,
  step = NULL,
  size = NULL,
  help = NULL,
  width = NULL
)
```

Arguments

`id`	Input id; value available as `input$id`.
`label`	Input label.
`value`	Initial value.
`...`	Extra attributes applied to the `<input>` element.
`min, max, step`	Numeric bounds and step.
`size`	Control size: `"sm"` or `"lg"`.
`help`	Help text rendered below the control (`.form-text`).
`width`	CSS width (e.g. `"100%"`, `"200px"`).

Value

A form control tag.

See Also

`shiny::numericInput()`

bs_offcanvas *Bootstrap offcanvas*

Description

A hidden sidebar panel that slides in from an edge of the viewport. The open/closed state is reported to the server as `input$id` (`TRUE` when shown), and can be driven server-side with `show_bs_offcanvas()`, `hide_bs_offcanvas()` and `toggle_bs_offcanvas()`. Open it declaratively from the UI with `bs_offcanvas_trigger()`.

Usage

```
bs_offcanvas(
  id,
  ...,
  title = NULL,
  placement = "start",
  backdrop = TRUE,
  scroll = FALSE,
  responsive = NULL,
  class = NULL
)
```

Arguments

`id`	Offcanvas id; open state available as `input$id`.
`...`	Offcanvas body content and named HTML attributes applied to the root.
`title`	Optional header title. When supplied, an `.offcanvas-header` with a title and a close button is rendered.
`placement`	Edge the panel slides in from: `"start"` (left), `"end"` (right), `"top"` or `"bottom"`.
`backdrop`	Backdrop behaviour: `TRUE` (default, dismiss on outside click), `FALSE` (no backdrop) or `"static"` (backdrop that does not dismiss on outside click).
`scroll`	If `TRUE`, allow body scrolling while the offcanvas is open.
`responsive`	Optional breakpoint (`"sm"`, `"md"`, `"lg"`, `"xl"`, `"xxl"`). Below it the panel behaves as an offcanvas; at or above it the content is shown inline (Bootstrap 5.2 responsive offcanvas).
`class`	Extra classes.

Value

An offcanvas tag.

Examples

```
bs_offcanvas("menu", "Sidebar content.", title = "Menu")
bs_offcanvas("nav", "Shown inline on lg+.", responsive = "lg")
```

bs_offcanvas_trigger *Trigger an offcanvas from the UI*

Description

Renders a button that opens the offcanvas whose id matches `target`, using Bootstrap's declarative `data-bs-toggle="offcanvas"` attributes (no server round trip required).

Usage

```
bs_offcanvas_trigger(target, ..., class = NULL)
```

Arguments

`target`	Id of the `bs_offcanvas()` to open.
`...`	Content (label) and named HTML attributes, forwarded to `bs_button()`.
`class`	Extra classes.

Value

A button tag.

Examples

```
bs_offcanvas_trigger("menu", "Open menu")
```

bs_page *A Bootstrap 5 page*

Description

Thin wrappers over `bslib::page()` / `bslib::page_fluid()` pinned to Bootstrap 5 that wire in the bootstrict dependency and default theme. Use these as the outermost call of a Shiny UI.

Usage

```
bs_page(
  ...,
  title = NULL,
  theme = bootstrict_theme(),
  color_mode = NULL,
  lang = "en"
)

bs_page_fluid(
```

```
  ...,
  title = NULL,
  theme = bootstrict_theme(),
  color_mode = NULL,
  lang = "en"
)

bs_page_fillable(
  ...,
  title = NULL,
  theme = bootstrict_theme(),
  color_mode = NULL,
  lang = "en"
)
```

Arguments

`...`	UI elements, and named HTML attributes for the page body.
`title`	Page title (browser tab).
`theme`	A `bootstrict_theme()` / `bslib::bs_theme()` object. Defaults to a stock Bootstrap 5 theme.
`color_mode`	Initial Bootstrap colour mode (`data-bs-theme` on the page body): `"light"` or `"dark"`. Switch it from the server with `set_bs_color_mode()`.
`lang`	Document language (`<html lang>`).

Value

A UI definition.

Examples

```
if (interactive()) {
  bs_page(
    theme = bootstrict_theme(primary = "#ff6600"),
    bs_container(bs_card(bs_card_body("Hello")))
  )
}
```

---

bs_page_item *Bootstrap pagination item*

Description

A single `<li class="page-item">` carrying a `.page-link` anchor. Use inside `bs_pagination()`.

Usage

```
bs_page_item(..., href = "#", active = FALSE, disabled = FALSE, class = NULL)
```

Arguments

`...`	Link content (text or tags) and named HTML attributes applied to the `<a class="page-link">`.
`href`	Link target.
`active`	If `TRUE`, mark as the current page (`.active`, `aria-current="page"`).
`disabled`	If `TRUE`, render as a non-interactive link (`.disabled`, `tabindex="-1"`, `aria-disabled="true"`).
`class`	Extra classes for the `<li>`.

Value

A `<li>` page-item tag.

Examples

```
bs_page_item("1", href = "#", active = TRUE)
```

---

bs_pagination *Bootstrap pagination*

Description

A list of page links, built from `bs_page_item()`s. For a quick numbered pager use `bs_pagination_numbered()`.

Usage

```
bs_pagination(
  ...,
  size = NULL,
  align = NULL,
  label = "Page navigation",
  class = NULL
)
```

Arguments

`...`	Page items built with `bs_page_item()`, plus named HTML attributes applied to the `<ul>`.
`size`	Size modifier: `"sm"` or `"lg"` (`.pagination-sm`/`.pagination-lg`).
`align`	Horizontal alignment of the pager: `"start"`, `"center"` or `"end"` (maps to `.justify-content-*`).
`label`	Accessible label for the surrounding `<nav>` (`aria-label`).
`class`	Extra classes for the `<ul>`.

Value

A `<nav>` pagination tag.

Examples

```
bs_pagination(
  bs_page_item("Previous", href = "#"),
  bs_page_item("1", href = "#", active = TRUE),
  bs_page_item("2", href = "#"),
  bs_page_item("Next", href = "#")
)
```

`bs_pagination_numbered` *Build a numbered pager*

Description

Convenience wrapper around `bs_pagination()` that lays out a "Previous" control, page numbers `1..n` (the `current` one active) and a "Next" control.

Usage

```
bs_pagination_numbered(
  n,
  current = 1,
  ...,
  href_template = NULL,
  size = NULL,
  align = NULL,
  label = "Page navigation",
  class = NULL
)
```

Arguments

`n`	Total number of pages.
`current`	Currently active page number (1-based).
`...`	Additional named HTML attributes forwarded to `bs_pagination()`'s `<ul>`.
`href_template`	Optional `sprintf()`-style template used to build each page's `href` from its page number, e.g. `"?page=%d"`. When `NULL`, every link uses `"#"`.
`size`	Size modifier passed to `bs_pagination()`.
`align`	Alignment passed to `bs_pagination()`.
`label`	Accessible label passed to `bs_pagination()`.
`class`	Extra classes for the `<ul>`.

Value

A `<nav>` pagination tag.

Examples

```
bs_pagination_numbered(5, current = 2)
```

`bs_password_input` *Bootstrap password input*

Description

Bootstrap password input

Usage

```
bs_password_input(
  id,
  label = NULL,
  value = "",
  ...,
  size = NULL,
  help = NULL,
  width = NULL
)
```

Arguments

`id`	Input id; value available as `input$id`.
`label`	Input label.
`value`	Initial value.
`...`	Extra attributes applied to the `<input>` element.
`size`	Control size: `"sm"` or `"lg"`.
`help`	Help text rendered below the control (`.form-text`).
`width`	CSS width (e.g. `"100%"`, `"200px"`).

Value

A form control tag.

See Also

`shiny::passwordInput()`

bs_placeholder *Bootstrap placeholder*

Description

Loading placeholders ("skeletons") that mimic the shape of content while it loads. Use `bs_placeholder()` for an individual placeholder and wrap one or more in `bs_placeholder_glow()` or `bs_placeholder_wave()` to animate them.

Usage

```
bs_placeholder(..., width = NULL, color = NULL, size = NULL, class = NULL)

bs_placeholder_glow(..., class = NULL)

bs_placeholder_wave(..., class = NULL)
```

Arguments

`...`	Additional named HTML attributes (and, for the wrappers, child content) applied to the element.
`width`	Column count (integer 1-12) controlling the placeholder width via the grid (`.col-*`).
`color`	Background theme colour (`.bg-*`), one of the Bootstrap theme colours.
`size`	Placeholder size, one of `"lg"`, `"sm"` or `"xs"`; `NULL` for the default size.
`class`	Extra classes.

Value

A placeholder tag.

Examples

```
bs_placeholder_glow(bs_placeholder(width = 6))
```

bs_popover *Add a Bootstrap popover to a UI element*

Description

Decorates an existing tag with the data attributes Bootstrap needs to render a popover. Popovers are initialised client-side by bootstrict (Bootstrap does not auto-initialise them).

Usage

```
bs_popover(
  tag,
  content,
  ...,
  title = NULL,
  placement = "right",
  trigger = "click",
  html = FALSE
)
```

Arguments

tag	A UI element (a shiny.tag) to attach the popover to.
content	Popover body content (or HTML, when html = TRUE).
...	Extra named attributes applied to tag.
title	Optional popover header.
placement	Popover placement: one of "top", "right", "bottom", "left".
trigger	How the popover is triggered (e.g. "click", "hover", "focus", "manual").
html	If TRUE, allow HTML content in the popover (data-bs-html).

Value

The decorated tag, with the bootstrict dependency attached.

Examples

```
bs_popover(shiny::tags$button("Click me"), "Popover body", title = "Heads up")
```

---

bs_progress *Bootstrap progress*

Description

A progress display. bs_progress_bar() builds one bar (a .progress track with its .progress-bar); bs_progress() finalises it, or stacks several bars into a Bootstrap 5.3 .progress-stacked group. Progress is display-only (it reports no value to the server); drive it from the server with update_bs_progress().

Usage

```
bs_progress(..., height = NULL, class = NULL)

bs_progress_bar(
  value = 0,
  ...,
```

```
  min = 0,
  max = 100,
  color = NULL,
  striped = FALSE,
  animated = FALSE,
  label = NULL,
  aria_label = NULL,
  id = NULL,
  class = NULL
)
```

Arguments

...	One or more bs_progress_bar()s and named HTML attributes. With two or more bars, the group renders as .progress-stacked.
height	CSS height of the progress track(s) (e.g. "20px", "1rem").
class	Extra classes.
value	Current value of the bar.
min, max	Lower and upper bounds of the scale.
color	Bar theme colour (.bg-*), one of the Bootstrap theme colours.
striped	If TRUE, apply the striped variant (.progress-bar-striped).
animated	If TRUE, animate the stripes (.progress-bar-animated).
label	Text shown inside the bar.
aria_label	Accessible name of the progress track (aria-label).
id	Id of the .progress track; required to target it with update_bs_progress().

Value

A progress tag.

Examples

```
bs_progress(bs_progress_bar(value = 25, id = "load"))
# stacked (Bootstrap 5.3):
bs_progress(
  bs_progress_bar(value = 15, color = "success"),
  bs_progress_bar(value = 30, color = "danger")
)
bs_progress_bar(value = 75, color = "success", striped = TRUE)
```

`bs_radio_input` *Bootstrap radio button group*

Description

Delegates to `shiny::radioButtons()` and rewrites the markup to Bootstrap 5 `.form-check` shape so the value stays available as `input$id`.

Usage

```
bs_radio_input(
  id,
  label = NULL,
  choices,
  selected = NULL,
  ...,
  inline = FALSE,
  reverse = FALSE,
  help = NULL,
  width = NULL
)
```

Arguments

`id`	Input id; selected value available as `input$id`.
`label`	Input label.
`choices`	Named or unnamed vector / list of choices.
`selected`	Initially selected value.
`...`	Extra attributes applied to each radio `<input>`.
`inline`	If `TRUE`, lay the choices out horizontally.
`reverse`	If `TRUE`, put the label before the control (`.form-check-reverse`, Bootstrap 5.2).
`help`	Help text rendered below the control (`.form-text`).
`width`	CSS width (e.g. `"100%"`, `"200px"`).

Value

A form control tag.

See Also

`shiny::radioButtons()`

Examples

```
bs_radio_input("size", "Size", c("S", "M", "L"), selected = "M")
```

bs_range_input *Bootstrap range (slider) input*

Description

A native `<input type="range" class="form-range">` whose value is reported to the server as `input$id`. Drive it server-side with `update_bs_range()`.

Usage

```
bs_range_input(
  id,
  label = NULL,
  value = NULL,
  min = 0,
  max = 100,
  step = NULL,
  ...,
  help = NULL,
  width = NULL
)
```

Arguments

`id`	Input id; value available as `input$id`.
`label`	Input label.
`value`	Initial value. If `NULL`, the browser initializes the control at its midpoint (`(min + max) / 2`), which is what `input$id` reports.
`min`, `max`, `step`	Numeric bounds and step.
`...`	Extra attributes applied to the `<input>` element.
`help`	Help text rendered below the control (`.form-text`).
`width`	CSS width (e.g. `"100%"`, `"200px"`).

Value

A form control tag.

Examples

```
bs_range_input("vol", "Volume", value = 50, min = 0, max = 100)
```

bs_ratio *Fixed aspect-ratio container*

Description

Wraps an embedded element (image, iframe, video, ...) so it keeps a fixed aspect ratio (`.ratio`).

Usage

```
bs_ratio(..., ratio = "16x9", class = NULL)
```

Arguments

`...`	The embedded element and named HTML attributes.
`ratio`	Aspect ratio: one of `"1x1"`, `"4x3"`, `"16x9"`, `"21x9"`.
`class`	Extra classes.

Value

A `<div>` tag.

Examples

```
bs_ratio(shiny::tags$iframe(src = "https://example.com"), ratio = "16x9")
```

bs_row *Bootstrap grid row*

Description

Bootstrap grid row

Usage

```
bs_row(
  ...,
  cols = NULL,
  gutters = NULL,
  gx = NULL,
  gy = NULL,
  justify = NULL,
  align = NULL,
  class = NULL
)
```

Arguments

...	Columns (bs_col()) and named HTML attributes.
cols	Number of equal-width columns per row (row-cols-*). A single value applies at all breakpoints; a named list (e.g. list(sm = 1, md = 2)) sets per-breakpoint counts.
gutters, gx, gy	Gutter width 0-5. gutters sets both axes; gx/gy override the horizontal / vertical gutter.
justify	Horizontal alignment of columns: "start", "center", "end", "around", "between", "evenly".
align	Vertical alignment of columns: "start", "center", "end".
class	Extra classes.

Value

A row tag.

Examples

```
bs_row(bs_col("a"), bs_col("b"), gutters = 3)
```

---

bs_scrollspy *Bootstrap scrollspy container*

Description

Wraps content in a scrollable region whose nav (identified by target) is updated as the user scrolls.

Usage

```
bs_scrollspy(
  target,
  ...,
  id = NULL,
  offset = NULL,
  smooth = TRUE,
  class = NULL
)
```

Arguments

target	Id (without #) of the nav / list-group that scrollspy drives.
...	Scrollable content and named HTML attributes.
id	Optional container id. The href of the currently active nav link is reported as input$id. Defaults to a unique auto-generated id (an id is required for bootstrict to initialise scrollspy, including inside renderUI() — Bootstrap only auto-initialises on full page load).

offset	Pixels from the top to offset link activation (data-bs-offset).
smooth	If TRUE, enable smooth scrolling (data-bs-smooth-scroll).
class	Extra classes.

Value

A scrollspy container tag, with the bootstrict dependency attached.

Examples

```
bs_scrollspy("nav-menu", shiny::tags$h4("Section"), offset = 100)
```

---

bs_select_input *Bootstrap select input*

Description

Renders a native Bootstrap 5 <select class="form-select"> (selectize is disabled to stay faithful to the Bootstrap markup).

Usage

```
bs_select_input(
  id,
  label = NULL,
  choices = NULL,
  selected = NULL,
  ...,
  multiple = FALSE,
  size = NULL,
  help = NULL,
  width = NULL
)
```

Arguments

id	Input id; value available as input$id.
label	Input label.
choices	Named or unnamed vector / list of choices.
selected	Initially selected value(s).
...	Extra attributes applied to the <input> element.
multiple	Allow multiple selection.
size	Control size: "sm" or "lg".
help	Help text rendered below the control (.form-text).
width	CSS width (e.g. "100%", "200px").

Value

A form control tag.

See Also

shiny::selectInput()

Examples

```
bs_select_input("fruit", "Fruit", c("Apple", "Pear"))
```

---

bs_spinner *Bootstrap spinner*

Description

An animated loading indicator. Render either a spinning border (`type = "border"`) or a pulsing dot (`type = "grow"`), optionally tinted with a theme colour and shrunk to the small variant.

Usage

```
bs_spinner(
  type = c("border", "grow"),
  color = NULL,
  size = NULL,
  label = "Loading...",
  ...,
  class = NULL
)
```

Arguments

`type`	Spinner style: `"border"` (default) or `"grow"`.
`color`	Theme colour (`.text-*`), one of the Bootstrap theme colours.
`size`	Spinner size; only `"sm"` (small) is accepted, `NULL` for the default size.
`label`	Visually hidden text announced to assistive technology.
`...`	Additional named HTML attributes applied to the spinner element.
`class`	Extra classes.

Value

A spinner tag.

Examples

```
bs_spinner(type = "border", color = "primary")
```

bs_table *Bootstrap table*

Description

Render a faithful Bootstrap 5 <table class="table">. Pass a data frame or matrix in data to have the header and body built automatically from its column names and rows; otherwise supply <thead>/<tbody> markup (or any table children) through ....

Usage

```
bs_table(
  data = NULL,
  ...,
  striped = FALSE,
  bordered = FALSE,
  borderless = FALSE,
  hover = FALSE,
  small = FALSE,
  variant = NULL,
  responsive = FALSE,
  align = NULL,
  caption = NULL,
  class = NULL
)
```

Arguments

data	A data frame or matrix to render. When NULL, the unnamed ... arguments are used as the table's children instead.
...	Manual table children (when data is NULL) and named HTML attributes applied to the <table> element.
striped	Add zebra-striping to table rows (.table-striped).
bordered	Add borders on all sides (.table-bordered).
borderless	Remove all borders (.table-borderless).
hover	Enable a hover state on rows (.table-hover).
small	Make the table more compact (.table-sm).
variant	Theme colour for the whole table (.table-*), one of the Bootstrap theme colours.
responsive	Make the table scroll horizontally on small devices. Use TRUE for .table-responsive, or a breakpoint string ("sm", "md", "lg", "xl", "xxl") for .table-responsive-{bp}.
align	Vertical alignment of cells (.align-*), e.g. "middle", "top", "bottom".
caption	Optional table caption text rendered in a <caption>.
class	Extra classes.

Value

A table tag (wrapped in a responsive container when `responsive` is set).

Examples

```
bs_table(head(mtcars), striped = TRUE, hover = TRUE)
```

`bs_tabset` *Bootstrap tabset*

Description

An interactive set of tabbed panels. The `data-value` of the currently shown panel is reported to the server as `input$id`, and the active tab can be driven server-side with `update_bs_tabset()`.

Usage

```
bs_tabset(
  id,
  ...,
  type = "tabs",
  selected = NULL,
  fill = FALSE,
  justified = FALSE,
  vertical = FALSE,
  class = NULL
)

bs_tab_panel(title, ..., value = NULL, icon = NULL, class = NULL)
```

Arguments

`id`	Tabset id; the active panel value is available as `input$id`.
`...`	Panels built with `bs_tab_panel()`.
`type`	Visual style: `"tabs"` (default), `"pills"` or `"underline"` (Bootstrap 5.3).
`selected`	Value of the panel shown initially (defaults to the first).
`fill`	If `TRUE`, tabs expand to fill available width (`.nav-fill`).
`justified`	If `TRUE`, tabs get equal width (`.nav-justified`).
`vertical`	If `TRUE`, lay tabs out vertically beside the content.
`class`	Extra classes for the wrapper.
`title`	Tab label content.
`value`	Panel identifier reported to the server (defaults to `title`).
`icon`	Optional icon placed before the title.

Value

A tabset tag.

Examples

```
bs_tabset(
  "tabs",
  bs_tab_panel("Home", "Home content", value = "home"),
  bs_tab_panel("Profile", "Profile content", value = "profile"),
  selected = "profile"
)
```

`bs_text_input` *Bootstrap text input*

Description

Bootstrap text input

Usage

```
bs_text_input(
  id,
  label = NULL,
  value = "",
  ...,
  placeholder = NULL,
  size = NULL,
  help = NULL,
  width = NULL
)
```

Arguments

`id`	Input id; value available as `input$id`.
`label`	Input label.
`value`	Initial value.
`...`	Extra attributes applied to the `<input>` element.
`placeholder`	Placeholder text.
`size`	Control size: `"sm"` or `"lg"`.
`help`	Help text rendered below the control (`.form-text`).
`width`	CSS width (e.g. `"100%"`, `"200px"`).

Value

A form control tag.

See Also

shiny::textInput()

Examples

```
bs_text_input("name", "Your name", placeholder = "Jane Doe")
```

`bs_textarea_input` *Bootstrap textarea input*

Description

Bootstrap textarea input

Usage

```
bs_textarea_input(
  id,
  label = NULL,
  value = "",
  ...,
  placeholder = NULL,
  rows = NULL,
  cols = NULL,
  size = NULL,
  help = NULL,
  width = NULL
)
```

Arguments

`id`	Input id; value available as `input$id`.
`label`	Input label.
`value`	Initial value.
`...`	Extra attributes applied to the `<input>` element.
`placeholder`	Placeholder text.
`rows, cols`	Visible rows / columns.
`size`	Control size: `"sm"` or `"lg"`.
`help`	Help text rendered below the control (`.form-text`).
`width`	CSS width (e.g. `"100%"`, `"200px"`).

Value

A form control tag.

See Also

shiny::textAreaInput()

---

bs_toast *Bootstrap toast*

Description

A lightweight, dismissible notification. The toast's visibility is reported to the server as `input$id` (`TRUE` when shown), and can be driven server-side with `show_bs_toast()` / `hide_bs_toast()`. Place one or more toasts inside a `bs_toast_container()` to position them on screen, or push transient notifications entirely from the server with `bs_notify_toast()`.

Usage

```
bs_toast(
  id,
  ...,
  title = NULL,
  icon = NULL,
  autohide = TRUE,
  delay = 5000,
  animation = TRUE,
  class = NULL
)
```

Arguments

`id`	Toast id; shown state available as `input$id`.
`...`	Toast body content and named HTML attributes applied to the root.
`title`	Optional header title. When supplied, a `.toast-header` with the title and a close button is rendered.
`icon`	Optional icon placed before the title in the header.
`autohide`	If `TRUE` (default), hide the toast automatically after `delay` milliseconds.
`delay`	Delay in milliseconds before auto-hiding (when `autohide`).
`animation`	If `FALSE`, disable the fade animation.
`class`	Extra classes.

Value

A toast tag.

Examples

```
bs_toast("hello", "Hello, world!", title = "Bootstrict")
```

bs_toast_container *Position toasts on screen*

Description

A fixed-position container that holds and lays out one or more `bs_toast()`s.

Usage

```
bs_toast_container(..., placement = "top-end", class = NULL)
```

Arguments

`...`	Toasts and named HTML attributes applied to the container.
`placement`	Where to anchor the container. One of `"top-start"`, `"top-center"`, `"top-end"`, `"middle-start"`, `"middle-center"`, `"middle-end"`, `"bottom-start"`, `"bottom-center"` or `"bottom-end"`.
`class`	Extra classes.

Value

A toast container tag.

Examples

```
bs_toast_container(
  bs_toast("hello", "Hi there", title = "Greeting"),
  placement = "top-end"
)
```

bs_tooltip *Add a Bootstrap tooltip to a UI element*

Description

Decorates an existing tag with the data attributes Bootstrap needs to render a tooltip. Tooltips are initialised client-side by bootstrict (Bootstrap does not auto-initialise them).

Usage

```
bs_tooltip(tag, title, ..., placement = "top", html = FALSE, trigger = NULL)
```

Arguments

tag	A UI element (a shiny.tag) to attach the tooltip to.
title	Tooltip text (or HTML, when html = TRUE).
...	Extra named attributes applied to tag.
placement	Tooltip placement: one of "top", "right", "bottom", "left".
html	If TRUE, allow HTML content in the tooltip (data-bs-html).
trigger	How the tooltip is triggered (e.g. "hover focus", "click", "manual"); NULL uses the Bootstrap default.

Value

The decorated tag, with the bootstrict dependency attached.

Examples

```
bs_tooltip(shiny::tags$button("Hover me"), "Tooltip text")
```

---

bs_valid_feedback *Bootstrap validation feedback*

Description

Inline messages shown next to a control to convey its validation state. bs_valid_feedback() renders .valid-feedback; bs_invalid_feedback() renders .invalid-feedback.

Usage

```
bs_valid_feedback(..., class = NULL)

bs_invalid_feedback(..., class = NULL)
```

Arguments

...	Feedback content and named HTML attributes.
class	Extra classes.

Value

A div tag.

Examples

```
bs_valid_feedback("Looks good!")
bs_invalid_feedback("Please choose a username.")
```

bs_visually_hidden *Visually hidden text*

Description

Renders content available to assistive technologies but hidden from sighted users (`.visually-hidden`).

Usage

```
bs_visually_hidden(..., class = NULL)
```

Arguments

`...`	Content and named HTML attributes.
`class`	Extra classes.

Value

A tag.

Examples

```
bs_visually_hidden("Loading, please wait")
```

bs_vr *Vertical rule*

Description

A vertical divider (`.vr`), the vertical counterpart of <hr>.

Usage

```
bs_vr(class = NULL)
```

Arguments

`class`	Extra classes.

Value

A tag.

Examples

```
bs_vr()
```

parse_scss_variables *Parse a SASS/SCSS variable sheet into a named list*

Description

Reads a `_variables.scss` style file (the kind a designer exports) and extracts top-level `$name: value;` declarations into a named list suitable for passing to `bootstrict_theme()` or `bslib::bs_theme()`. Trailing `!default` / `!global` flags and line/block comments are stripped. Values are returned verbatim as strings (Sass resolves them at compile time), so maps, functions and colour expressions all pass straight through.

Usage

```
parse_scss_variables(path)
```

Arguments

`path`	Path to a `.scss` file (SCSS syntax, `$name: value;` — the indented `.sass` syntax has no semicolons and cannot be parsed).

Value

A named list of Sass variable values. Names use the Bootstrap convention without the leading `$` (e.g. `primary`, `font-family-base`).

Examples

```
tmp <- tempfile(fileext = ".scss")
writeLines(c("$primary: #ff6600;", "$border-radius: 0.5rem !default;"), tmp)
parse_scss_variables(tmp)
```

set_bs_color_mode *Switch the Bootstrap colour mode from the server*

Description

Sets the Bootstrap 5.3 colour mode (`data-bs-theme`) on the page body, switching every component between light and dark. Set the initial mode with the `color_mode` argument of `bs_page()`.

Usage

```
set_bs_color_mode(mode, session = shiny::getDefaultReactiveDomain())
```

Arguments

`mode`	`"light"` or `"dark"`.
`session`	The Shiny session.

Value

Invisibly NULL, called for its side effect.

Examples

```
if (interactive()) set_bs_color_mode("dark")
```

---

show_bs_modal *Control a modal from the server*

Description

Show, hide or toggle a bs_modal() from server code.

Usage

```
show_bs_modal(id, session = shiny::getDefaultReactiveDomain())

hide_bs_modal(id, session = shiny::getDefaultReactiveDomain())

toggle_bs_modal(id, session = shiny::getDefaultReactiveDomain())
```

Arguments

id	Modal id (namespaced automatically inside modules).
session	The Shiny session.

Value

Invisibly NULL, called for its side effect.

Examples

```
if (interactive()) show_bs_modal("info")
if (interactive()) hide_bs_modal("info")
if (interactive()) toggle_bs_modal("info")
```

show_bs_offcanvas *Control an offcanvas from the server*

Description

Show, hide or toggle a `bs_offcanvas()` from server code.

Usage

```
show_bs_offcanvas(id, session = shiny::getDefaultReactiveDomain())

hide_bs_offcanvas(id, session = shiny::getDefaultReactiveDomain())

toggle_bs_offcanvas(id, session = shiny::getDefaultReactiveDomain())
```

Arguments

id	Offcanvas id (namespaced automatically inside modules).
session	The Shiny session.

Value

Invisibly `NULL`, called for its side effect.

Examples

```
if (interactive()) show_bs_offcanvas("menu")
if (interactive()) hide_bs_offcanvas("menu")
if (interactive()) toggle_bs_offcanvas("menu")
```

show_bs_toast *Control a toast from the server*

Description

Show or hide a `bs_toast()` from server code.

Usage

```
show_bs_toast(id, session = shiny::getDefaultReactiveDomain())

hide_bs_toast(id, session = shiny::getDefaultReactiveDomain())
```

Arguments

id	Toast id (namespaced automatically inside modules).
session	The Shiny session.

Value

Invisibly NULL, called for its side effect.

Examples

```
if (interactive()) show_bs_toast("hello")
if (interactive()) hide_bs_toast("hello")
```

---

update_bs_accordion *Control an accordion from the server*

Description

Control an accordion from the server

Usage

```
update_bs_accordion(
  id,
  open = NULL,
  close = NULL,
  session = shiny::getDefaultReactiveDomain()
)
```

Arguments

id	Accordion id (namespaced automatically inside modules).
open	Panel value(s) to open. Use TRUE to open all (sensible with multiple = TRUE); FALSE/NULL is a no-op.
close	Panel value(s) to close. Use TRUE to close all; FALSE/NULL is a no-op.
session	The Shiny session.

Value

Invisibly NULL, called for its side effect.

update_bs_carousel *Control a carousel from the server*

Description

Control a carousel from the server

Usage

```
update_bs_carousel(
  id,
  to = NULL,
  slide = NULL,
  session = shiny::getDefaultReactiveDomain()
)
```

Arguments

id	Carousel id (namespaced automatically inside modules).
to	0-based index of the slide to cycle to. Takes precedence over `slide` when both are supplied.
slide	Direction to advance: `"next"` or `"prev"`.
session	The Shiny session.

Value

Invisibly `NULL`, called for its side effect.

Examples

```
## Not run:
update_bs_carousel("demo", to = 2)
update_bs_carousel("demo", slide = "next")

## End(Not run)
```

update_bs_collapse *Control a collapse from the server*

Description

Show, hide or toggle a `bs_collapse()` from server code.

Usage

```
update_bs_collapse(
  id,
  action = c("toggle", "show", "hide"),
  session = shiny::getDefaultReactiveDomain()
)
```

Arguments

id	Collapse id (namespaced automatically inside modules).
action	One of "toggle", "show" or "hide".
session	The Shiny session.

Value

Invisibly NULL, called for its side effect.

Examples

```
if (interactive()) update_bs_collapse("more", "show")
```

---

update_bs_color *Update a colour input from the server*

Description

Routes through shiny::session's sendInputMessage() so the colour binding's receiveMessage updates the swatch.

Usage

```
update_bs_color(id, value, session = shiny::getDefaultReactiveDomain())
```

Arguments

id	Colour input id (namespaced automatically inside modules).
value	New colour as a hex string (e.g. "#0d6efd").
session	The Shiny session.

Value

Invisibly NULL, called for its side effect.

Examples

```
if (interactive()) update_bs_color("col", "#198754")
```

update_bs_list_group *Control a list group selection from the server*

Description

Activates the item whose `value` matches `selected` in a selectable `bs_list_group()` (one created with an `id`).

Usage

```
update_bs_list_group(
  id,
  selected = NULL,
  session = shiny::getDefaultReactiveDomain()
)
```

Arguments

`id`	List group id (namespaced automatically inside modules).
`selected`	Item value to activate. `NULL` leaves the selection unchanged.
`session`	The Shiny session.

Value

Invisibly `NULL`, called for its side effect.

Examples

```
## Not run:
update_bs_list_group("my_group", selected = "two")

## End(Not run)
```

update_bs_progress *Update a progress bar from the server*

Description

Update a progress bar from the server

Usage

```
update_bs_progress(
  id,
  value = NULL,
  label = NULL,
  color = NULL,
  min = NULL,
  max = NULL,
  session = shiny::getDefaultReactiveDomain()
)
```

Arguments

id	Id of the bs_progress_bar() track to update (namespaced automatically inside modules).
value	New value.
label	New text shown inside the bar.
color	New theme colour (.bg-*).
min, max	New lower / upper bounds of the scale.
session	The Shiny session.

Value

Invisibly NULL, called for its side effect.

Examples

```
## Not run:
update_bs_progress("load", value = 80)

## End(Not run)
```

---

update_bs_range *Update a range input from the server*

Description

Routes through shiny::session's sendInputMessage() so the range binding's receiveMessage updates the slider (the "shiny-like" update path).

Usage

```
update_bs_range(id, value, session = shiny::getDefaultReactiveDomain())
```

Arguments

id	Range input id (namespaced automatically inside modules).
value	New value.
session	The Shiny session.

Value

Invisibly NULL, called for its side effect.

Examples

```
if (interactive()) update_bs_range("vol", 75)
```

---

update_bs_tabset *Control a tabset from the server*

Description

Control a tabset from the server

Usage

```
update_bs_tabset(id, selected, session = shiny::getDefaultReactiveDomain())
```

Arguments

id	Tabset id (namespaced automatically inside modules).
selected	Value of the panel to show.
session	The Shiny session.

Value

Invisibly NULL, called for its side effect.

Examples

```
## Not run:
update_bs_tabset("tabs", selected = "profile")

## End(Not run)
```

use_bootstrict *Activate bootstrict inside a UI*

Description

Returns the bootstrict HTML dependency (Shiny input bindings + supporting CSS) so it can be dropped anywhere in a UI. Page constructors such as `bs_page()` call this for you; use it directly when embedding bootstrict widgets into a UI you build by hand.

Usage

```
use_bootstrict()
```

Value

An htmltools::htmlDependency.

use_bootstrict_golem *Scaffold a bootstrict app as a golem project hook*

Description

A `golem::create_golem()` `project_hook` that turns a freshly created golem skeleton into a minimal bootstrict application:

Usage

```
use_bootstrict_golem(path, package_name, ...)
```

Arguments

`path`	Path of the newly created golem project. Unused, kept for compatibility with the golem hook interface.
`package_name`	Name of the package / application being created.
`...`	Reserved for compatibility with the golem hook interface.

Details

- `R/app_ui.R` is rewritten to use `bs_page()` and shows a "Hello world" inside a `bs_container()`.
- An (empty by default) `inst/app/www/_variables.scss` designer variable sheet is created and wired into `bootstrict_theme()`, so a designer can drop SCSS variables (`$name: value;`) in and have them picked up automatically.

golem sets the working directory to the new project before calling the hook, so every path below is relative to the application root.

Value

Invisibly NULL. Called for its side effects on the project files.

Examples

```
if (interactive()) {
  golem::create_golem(
    "my.app",
    project_hook = bootstrict::use_bootstrict_golem
  )
}
```

Index