

Package: shinyblocks (via r-universe)

September 22, 2026

Title Composable Modern UI Blocks for Shiny

Version 0.0.0.9003

Description Provides composable Shiny dashboard components inspired by shadcn/ui, with R-first helpers, package-local runtime assets, scoped theming, and accessible defaults. End users do not run Node, Tailwind, Vite, or frontend build tooling.

URL <https://nvelden.github.io/shinyblocks/>

License MIT + file LICENSE

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

Collate 'deps.R' 'utils.R' 'runtime-payload.R' 'runtime.R'
'runtime-input-update.R' 'icon.R' 'card.R' 'button.R'
'task-button.R' 'badge.R' 'code.R' 'table.R' 'image.R'
'alert.R' 'overlays.R' 'dropdown-menu.R' 'accordion.R'
'indicators.R' 'components.R' 'field.R' 'input-group.R'
'select.R' 'combobox.R' 'date-picker.R' 'date-range-picker.R'
'form-controls.R' 'radio-group.R' 'toggle-group.R'
'pagination.R' 'tabs.R' 'breadcrumb.R' 'toast.R'
'style-profiles.R' 'style.R' 'theme-presets.R' 'theme.R'
'dark-mode.R' 'layout.R' 'package.R' 'page.R' 'showcase.R'
'zzz.R'

Imports htmltools, jsonlite, rlang, shiny

Suggests testthat (>= 3.0.0), withr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

RoxygenNote 7.3.3

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

Repository <https://nvelden.r-universe.dev>

Date/Publication 2026-09-03 16:05:49 UTC

RemoteUrl <https://github.com/nvelden/shinyblocks>

RemoteRef HEAD

RemoteSha 720e4da703178af66f7582cd0c19b8ed55a4312a

Contents

block_accordion	4
block_accordion_item	6
block_alert	7
block_alert_action	8
block_alert_description	9
block_alert_dialog	9
block_alert_title	11
block_badge	12
block_body	13
block_breadcrumb	13
block_breadcrumb_ellipsis	14
block_breadcrumb_item	15
block_button	15
block_card	16
block_card_content	17
block_card_description	18
block_card_footer	19
block_card_header	19
block_card_title	20
block_checkbox	21
block_cluster	22
block_code	23
block_combobox	24
block_dark_mode_toggle	25
block_date_picker	26
block_date_range_picker	28
block_dialog	29
block_dropdown_menu	31
block_empty	32
block_field	33
block_field_description	34
block_field_group	35
block_field_invalid	35
block_field_label	36
block_field_legend	37
block_field_set	38
block_file_input	38
block_grid	40
block_header	41
block_icon	42
block_image_output	42

block_input	44
block_input_group	45
block_input_group_addon	46
block_nav	47
block_nav_group	47
block_nav_item	48
block_nav_label	49
block_page	50
block_pagination	51
block_plot_output	52
block_popover	53
block_progress	54
block_radio_group	56
block_select	57
block_separator	58
block_sidebar	59
block_skeleton	60
block_slider	60
block_spinner	62
block_stack	63
block_style	64
block_style_profiles	65
block_switch	65
block_tab	66
block_table	67
block_tabs	69
block_task_button	70
block_textarea	71
block_theme	72
block_theme_presets	73
block_toaster	74
block_toggle_group	75
block_tooltip	76
block_value_box	78
dismiss_toast	79
dropdown_menu_item	80
dropdown_menu_label	81
dropdown_menu_separator	81
inc_block_progress	82
run_showcase	83
show_toast	83
table_column	85
update_block_accordion	86
update_block_alert_dialog	87
update_block_button	89
update_block_checkbox	90
update_block_combobox	91
update_block_date_picker	92

update_block_date_range_picker	93
update_block_dialog	95
update_block_dropdown_menu	96
update_block_file_input	97
update_block_input	99
update_block_nav	100
update_block_pagination	101
update_block_popover	102
update_block_progress	103
update_block_radio_group	104
update_block_select	106
update_block_slider	107
update_block_switch	108
update_block_table	109
update_block_tabs	111
update_block_task_button	111
update_block_textarea	113
update_block_theme	114
update_block_toaster	115
update_block_toggle_group	116
Index	118

block_accordion	<i>Create an accordion</i>
-----------------	----------------------------

Description

A vertically stacked set of collapsible sections built from [block_accordion_item\(\)](#). Use it to organize long content into expand/collapse panels (FAQs, grouped settings, filters).

Usage

```
block_accordion(  
  ...,  
  id = NULL,  
  type = c("single", "multiple"),  
  collapsible = FALSE,  
  open = NULL,  
  style = NULL,  
  class = NULL  
)
```

Arguments

...	<code>block_accordion_item()</code> items. Supports <code>!!!</code> to splice a list of items.
<code>id</code>	Optional input id. When supplied, <code>input\$<id></code> reports the open item value(s).
<code>type</code>	"single" (at most one item open at a time) or "multiple" (any number open independently). Create-only.
<code>collapsible</code>	For <code>type = "single"</code> only: whether the open item can be collapsed, leaving nothing open. Ignored for <code>type = "multiple"</code> (always collapsible). Create-only.
<code>open</code>	Item value(s) to open initially. For <code>type = "single"</code> a single value or NULL; for <code>type = "multiple"</code> a character vector. Must match item values.
<code>style</code>	Inline CSS styles applied to the accordion wrapper.
<code>class</code>	Additional classes for the accordion wrapper.

Details

The trigger buttons carry `aria-expanded/aria-controls`, the chevron rotates on open, and panel height animates. When `id` is supplied the open item value(s) are reported to `input$<id>`: a string (or NULL) for `type = "single"`, a character vector for `type = "multiple"`.

... supports `rlang`'s `!!!` splice operator, so a programmatically built list of items (e.g. one per row of a data frame) can be passed without `do.call()`:

```
items <- lapply(seq_len(nrow(faqs)), function(i) {
  block_accordion_item(faqs$value[i], faqs$question[i], faqs$answer[i])
})
block_accordion(!!!items, id = "faq")
```

Value

An `htmltools` tag.

See Also

Other content: `block_accordion_item()`, `block_alert()`, `block_alert_action()`, `block_alert_description()`, `block_alert_dialog()`, `block_alert_title()`, `block_badge()`, `block_card()`, `block_card_content()`, `block_card_description()`, `block_card_footer()`, `block_card_header()`, `block_card_title()`, `block_code()`, `block_dialog()`, `block_dropdown_menu()`, `block_empty()`, `block_popover()`, `block_progress()`, `block_separator()`, `block_skeleton()`, `block_spinner()`, `block_table()`, `block_toaster()`, `block_tooltip()`, `block_value_box()`, `dismiss_toast()`, `dropdown_menu_item()`, `dropdown_menu_label()`, `dropdown_menu_separator()`, `inc_block_progress()`, `show_toast()`, `table_column()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`, `update_block_dropdown_menu()`, `update_block_popover()`, `update_block_progress()`, `update_block_table()`, `update_block_toaster()`

block_accordion_item *Create an accordion item*

Description

A single collapsible section for `block_accordion()`: a trigger button (the title) that expands or collapses its body content. Body content is arbitrary Shiny/htmltools markup and stays live in the DOM, so reactive outputs inside a closed panel keep working.

Usage

```
block_accordion_item(
  value,
  title,
  ...,
  icon = NULL,
  disabled = FALSE,
  class = NULL
)
```

Arguments

value	String identifying the item. Required and must be unique within an accordion. This is the value reported to <code>input\$<id></code> and the value <code>update_block_accordion()</code> opens or closes.
title	Trigger label. A single string, or an htmltools tag for richer content.
...	Panel body content (htmltools tags, Shiny outputs, ...).
icon	Optional leading icon shown before the title: a vendored icon name (see <code>block_icon()</code>) or an htmltools tag.
disabled	Whether the item is non-interactive (cannot be toggled).
class	Additional classes for the item wrapper.

Value

An accordion-item tag consumed by `block_accordion()`.

See Also

Other content: `block_accordion()`, `block_alert()`, `block_alert_action()`, `block_alert_description()`, `block_alert_dialog()`, `block_alert_title()`, `block_badge()`, `block_card()`, `block_card_content()`, `block_card_description()`, `block_card_footer()`, `block_card_header()`, `block_card_title()`, `block_code()`, `block_dialog()`, `block_dropdown_menu()`, `block_empty()`, `block_popover()`, `block_progress()`, `block_separator()`, `block_skeleton()`, `block_spinner()`, `block_table()`, `block_toaster()`, `block_tooltip()`, `block_value_box()`, `dismiss_toast()`, `dropdown_menu_item()`, `dropdown_menu_label()`, `dropdown_menu_separator()`, `inc_block_progress()`, `show_toast()`, `table_column()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`,

[update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

block_alert	Create an alert
-------------	-----------------

Description

Create an alert

Usage

```
block_alert(  
  title,  
  ...,  
  description = NULL,  
  action = NULL,  
  icon = "info",  
  variant = c("default", "destructive", "success", "warning", "info"),  
  class = NULL,  
  style = NULL  
)
```

Arguments

title	Alert title. Required for accessibility.
...	Additional alert body content.
description	Optional alert description.
action	Optional action content, such as a block_button() .
icon	Optional icon tag or vendored icon name.
variant	Visual variant.
class	Additional classes.
style	Optional inline custom styles.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#),

```
dropdown_menu_label(), dropdown_menu_separator(), inc_block_progress(), show_toast(),
table_column(), update_block_accordion(), update_block_alert_dialog(), update_block_dialog(),
update_block_dropdown_menu(), update_block_popover(), update_block_progress(), update_block_table(),
update_block_toaster()
```

block_alert_action	<i>Create an alert action</i>
--------------------	-------------------------------

Description

Create an alert action

Usage

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

Arguments

...	Alert action content, such as a <code>block_button()</code> .
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other content: `block_accordion()`, `block_accordion_item()`, `block_alert()`, `block_alert_description()`, `block_alert_dialog()`, `block_alert_title()`, `block_badge()`, `block_card()`, `block_card_content()`, `block_card_description()`, `block_card_footer()`, `block_card_header()`, `block_card_title()`, `block_code()`, `block_dialog()`, `block_dropdown_menu()`, `block_empty()`, `block_popover()`, `block_progress()`, `block_separator()`, `block_skeleton()`, `block_spinner()`, `block_table()`, `block_toaster()`, `block_tooltip()`, `block_value_box()`, `dismiss_toast()`, `dropdown_menu_item()`, `dropdown_menu_label()`, `dropdown_menu_separator()`, `inc_block_progress()`, `show_toast()`, `table_column()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`, `update_block_dropdown_menu()`, `update_block_popover()`, `update_block_progress()`, `update_block_table()`, `update_block_toaster()`

 block_alert_description

Create an alert description

Description

Create an alert description

Usage

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

Arguments

...	Alert description content.
class	Additional classes.

Value

An htmltools tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

 block_alert_dialog

Create an alert dialog

Description

A modal confirmation dialog that requires an explicit confirm or cancel choice. `input$<id>` is initially NULL and reports "confirm" or "cancel" for each outcome. Escape cancels; clicking the backdrop does nothing.

Usage

```

block_alert_dialog(
  id,
  title,
  description = NULL,
  ...,
  confirm_label = "Continue",
  cancel_label = "Cancel",
  trigger = NULL,
  open = FALSE,
  confirm_variant = c("default", "destructive"),
  size = c("default", "sm", "lg", "xl"),
  class = NULL,
  style = NULL
)

```

Arguments

<code>id</code>	Required input id.
<code>title</code>	Required accessible title.
<code>description</code>	Optional supporting description.
<code>...</code>	Optional body content, serialized to HTML.
<code>confirm_label</code>	Label for the confirmation action.
<code>cancel_label</code>	Label for the cancellation action.
<code>trigger</code>	Optional button label that opens the alert dialog.
<code>open</code>	Initial open state.
<code>confirm_variant</code>	Confirmation button variant: "default" or "destructive".
<code>size</code>	Content max-width preset. One of "sm", "default", "lg", or "xl".
<code>class</code>	Additional classes for the dialog content container.
<code>style</code>	Optional inline CSS for the dialog content container.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#),

`table_column()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`, `update_block_dropdown_menu()`, `update_block_popover()`, `update_block_progress()`, `update_block_table()`, `update_block_toaster()`

block_alert_title	Create an alert title
-------------------	-----------------------

Description

Create an alert title

Usage

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

Arguments

- `...` Alert title content.
- `class` Additional classes.

Value

An `htmltools` tag.

See Also

Other content: `block_accordion()`, `block_accordion_item()`, `block_alert()`, `block_alert_action()`, `block_alert_description()`, `block_alert_dialog()`, `block_badge()`, `block_card()`, `block_card_content()`, `block_card_description()`, `block_card_footer()`, `block_card_header()`, `block_card_title()`, `block_code()`, `block_dialog()`, `block_dropdown_menu()`, `block_empty()`, `block_popover()`, `block_progress()`, `block_separator()`, `block_skeleton()`, `block_spinner()`, `block_table()`, `block_toaster()`, `block_tooltip()`, `block_value_box()`, `dismiss_toast()`, `dropdown_menu_item()`, `dropdown_menu_label()`, `dropdown_menu_separator()`, `inc_block_progress()`, `show_toast()`, `table_column()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`, `update_block_dropdown_menu()`, `update_block_popover()`, `update_block_progress()`, `update_block_table()`, `update_block_toaster()`

block_badge	Create a badge
-------------	----------------

Description

Create a badge

Usage

```
block_badge(  
  label,  
  variant = c("default", "secondary", "outline", "destructive", "success", "warning",  
    "info", "ghost", "link"),  
  size = c("default", "sm", "lg"),  
  class = NULL,  
  style = NULL  
)
```

Arguments

label	Badge label.
variant	Visual variant.
size	Visual size.
class	Additional classes.
style	Optional inline custom styles.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

block_body	Create a page body landmark
------------	-----------------------------

Description

Create a page body landmark

Usage

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

Arguments

...	Body content.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other layout: [block_cluster\(\)](#), [block_grid\(\)](#), [block_header\(\)](#), [block_page\(\)](#), [block_sidebar\(\)](#), [block_stack\(\)](#)

block_breadcrumb	Create a breadcrumb trail
------------------	---------------------------

Description

Static navigation landmark showing the path to the current page, following the shaded breadcrumb pattern. Children must be [block_breadcrumb_item\(\)](#) or [block_breadcrumb_ellipsis\(\)](#) entries; a separator is inserted between consecutive children automatically and hidden from assistive technology.

Usage

```
block_breadcrumb(..., separator = NULL, style = NULL, class = NULL)
```

Arguments

...	Breadcrumb entries built with block_breadcrumb_item() or block_breadcrumb_ellipsis() .
separator	Optional separator rendered between entries. Either a string (e.g. <code>"/"</code>) or an <code>htmltools</code> tag; defaults to a chevron icon.
style	Inline CSS styles for the <code><nav></code> container.
class	Additional classes for the <code><nav></code> container.

Value

An `htmltools` tag: a `<nav aria-label="breadcrumb">` landmark.

See Also

Other navigation: [block_breadcrumb_ellipsis\(\)](#), [block_breadcrumb_item\(\)](#), [block_nav\(\)](#), [block_nav_group\(\)](#), [block_nav_item\(\)](#), [block_nav_label\(\)](#), [block_tab\(\)](#), [block_tabs\(\)](#), [update_block_nav\(\)](#), [update_block_tabs\(\)](#)

`block_breadcrumb_ellipsis`

Create a collapsed-middle breadcrumb marker

Description

Placeholder for hidden entries in a long trail, following `shadcn's BreadcrumbEllipsis`: a decorative ellipsis icon hidden from assistive technology with a visually hidden text alternative.

Usage

```
block_breadcrumb_ellipsis(label = "More", class = NULL)
```

Arguments

<code>label</code>	Visually hidden text announced to assistive technology.
<code>class</code>	Additional classes for the <code></code> entry.

Value

An `htmltools` tag.

See Also

Other navigation: [block_breadcrumb\(\)](#), [block_breadcrumb_item\(\)](#), [block_nav\(\)](#), [block_nav_group\(\)](#), [block_nav_item\(\)](#), [block_nav_label\(\)](#), [block_tab\(\)](#), [block_tabs\(\)](#), [update_block_nav\(\)](#), [update_block_tabs\(\)](#)

block_breadcrumb_item	Create a breadcrumb entry
-----------------------	---------------------------

Description

Create a breadcrumb entry

Usage

```
block_breadcrumb_item(label, href = NULL, current = FALSE, class = NULL)
```

Arguments

label	Entry label. A string or an <code>htmltools</code> tag.
href	Destination URL. Ignored when <code>current = TRUE</code> (the current page is not a link).
current	Whether this entry is the current page. Rendered as a non-interactive <code></code> .
class	Additional classes for the <code></code> entry.

Value

An `htmltools` tag.

See Also

Other navigation: [block_breadcrumb\(\)](#), [block_breadcrumb_ellipsis\(\)](#), [block_nav\(\)](#), [block_nav_group\(\)](#), [block_nav_item\(\)](#), [block_nav_label\(\)](#), [block_tab\(\)](#), [block_tabs\(\)](#), [update_block_nav\(\)](#), [update_block_tabs\(\)](#)

block_button	Create a modern button
--------------	------------------------

Description

Create a modern button

Usage

```
block_button(  
  label,  
  variant = c("default", "secondary", "outline", "ghost", "destructive", "link"),  
  size = c("default", "sm", "lg", "icon"),  
  icon = NULL,  
  icon_position = c("inline-start", "inline-end"),  
  ...,  
  class = NULL  
)
```

Arguments

label	Button label.
variant	Visual variant.
size	Button size.
icon	Optional icon tag or vendored icon name.
icon_position	Whether the icon appears before or after the label.
...	Additional attributes passed to the rendered button. Pass <code>id = "..."</code> here to make the button addressable via update_block_button() . Runtime-owned type, data-slot, variant/size, class, style, and disabled attributes cannot be overridden here.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other action: [block_task_button\(\)](#), [update_block_button\(\)](#), [update_block_task_button\(\)](#)

block_card	<i>Create a dashboard card</i>
------------	--------------------------------

Description

Create a dashboard card

Usage

```
block_card(  
  ...,  
  title = NULL,  
  description = NULL,  
  value = NULL,  
  footer = NULL,  
  class = NULL,  
  style = NULL  
)
```

Arguments

...	Card body content or composed card region tags.
title	Optional card title.
description	Optional card description.
value	Optional primary value.
footer	Optional card footer content.
class	Additional classes.
style	Optional inline custom styles.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

block_card_content	<i>Create card content</i>
--------------------	----------------------------

Description

Create card content

Usage

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

Arguments

...	Card content.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other content: `block_accordion()`, `block_accordion_item()`, `block_alert()`, `block_alert_action()`, `block_alert_description()`, `block_alert_dialog()`, `block_alert_title()`, `block_badge()`, `block_card()`, `block_card_description()`, `block_card_footer()`, `block_card_header()`, `block_card_title()`, `block_code()`, `block_dialog()`, `block_dropdown_menu()`, `block_empty()`, `block_popover()`, `block_progress()`, `block_separator()`, `block_skeleton()`, `block_spinner()`, `block_table()`, `block_toaster()`, `block_tooltip()`, `block_value_box()`, `dismiss_toast()`, `dropdown_menu_item()`, `dropdown_menu_label()`, `dropdown_menu_separator()`, `inc_block_progress()`, `show_toast()`, `table_column()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`, `update_block_dropdown_menu()`, `update_block_popover()`, `update_block_progress()`, `update_block_table()`, `update_block_toaster()`

`block_card_description`

Create a card description

Description

Create a card description

Usage

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

Arguments

<code>...</code>	Card description content.
<code>class</code>	Additional classes.

Value

An `htmltools` tag.

See Also

Other content: `block_accordion()`, `block_accordion_item()`, `block_alert()`, `block_alert_action()`, `block_alert_description()`, `block_alert_dialog()`, `block_alert_title()`, `block_badge()`, `block_card()`, `block_card_content()`, `block_card_footer()`, `block_card_header()`, `block_card_title()`, `block_code()`, `block_dialog()`, `block_dropdown_menu()`, `block_empty()`, `block_popover()`, `block_progress()`, `block_separator()`, `block_skeleton()`, `block_spinner()`, `block_table()`, `block_toaster()`, `block_tooltip()`, `block_value_box()`, `dismiss_toast()`, `dropdown_menu_item()`, `dropdown_menu_label()`, `dropdown_menu_separator()`, `inc_block_progress()`, `show_toast()`, `table_column()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`, `update_block_dropdown_menu()`, `update_block_popover()`, `update_block_progress()`, `update_block_table()`, `update_block_toaster()`

block_card_footer	Create a card footer
-------------------	----------------------

Description

Create a card footer

Usage

block_card_footer(..., class = NULL)

Arguments

- ... Card footer content.
- class Additional classes.

Value

An htmltools tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

block_card_header	Create a card header
-------------------	----------------------

Description

Create a card header

Usage

block_card_header(..., class = NULL)

Arguments

- ... Card header content.
- class Additional classes.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

<code>block_card_title</code>	<i>Create a card title</i>
-------------------------------	----------------------------

Description

Create a card title

Usage

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

Arguments

- ... Card title content.
- class Additional classes.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

block_checkbox	Create a styled checkbox input
----------------	--------------------------------

Description

Create a styled checkbox input

Usage

```
block_checkbox(
  input_id,
  label,
  value = FALSE,
  disabled = FALSE,
  style = NULL,
  class = NULL
)
```

Arguments

input_id	Input id.
label	Checkbox label.
value	Whether the checkbox starts checked.
disabled	Whether the control is disabled.
style	Inline CSS styles.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other forms: `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

block_cluster	<i>Cluster content horizontally</i>
---------------	-------------------------------------

Description

Arrange content in a horizontal group with semantic spacing and optional wrapping.

Usage

```
block_cluster(
  ...,
  gap = c("sm", "md", "lg"),
  align = c("center", "start", "end", "stretch"),
  justify = c("start", "center", "end", "between"),
  wrap = TRUE,
  class = NULL
)
```

Arguments

...	Child content to arrange. Named arguments are applied to the container as HTML attributes (the <code>htmltools</code> convention), e.g. <code>id</code> , <code>style</code> , or <code>data-*</code> . Layout itself (display, wrapping, gap, alignment) is owned by the primitive's classes — use <code>gap/align/justify/wrap</code> , not an inline style, to control it.
gap	Spacing between children: "sm", "md", or "lg".
align	Cross-axis alignment: "center", "start", "end", or "stretch".
justify	Main-axis distribution: "start", "center", "end", or "between".
wrap	Whether children may wrap onto additional rows.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other layout: [block_body\(\)](#), [block_grid\(\)](#), [block_header\(\)](#), [block_page\(\)](#), [block_sidebar\(\)](#), [block_stack\(\)](#)

 block_code

Create a code block

Description

A pre-formatted code block following the shadcn documentation code surface: bordered frame, monospace text, optional line numbers, and an optional copy-to-clipboard button.

Usage

```
block_code(
  code,
  language = NULL,
  copyable = TRUE,
  line_numbers = TRUE,
  header = FALSE,
  variant = c("default", "outline"),
  class = NULL,
  style = NULL,
  ...
)
```

Arguments

code	The code string to display.
language	Optional programming language name to display when header = TRUE.
copyable	Logical. If TRUE (default), displays a copy-to-clipboard button.
line_numbers	Logical. If TRUE (default), displays line numbers.
header	Logical. If TRUE, displays an optional header with editor dots and the language label. Defaults to FALSE to match the shadcn documentation examples.
variant	Visual variant. One of "default" (background surface) or "outline" (transparent surface).
class	Additional CSS classes.
style	Optional inline custom styles.
...	Additional attributes or child elements (passed down).

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

 block_combobox

Create a searchable select (combobox) input

Description

`block_combobox()` renders a shadcn-style combobox: a trigger plus a portal-rendered popup whose first row is a type-to-filter search box over the choices. Like [block_select\(\)](#) it is backed by a hidden native `<select>` that carries the Shiny input value, and it supports single and multiple selection. Reach for it over [block_select\(\)](#) when the choice list is long enough that users benefit from filtering (the searchable-select gap that otherwise pushes people toward heavier third-party dropdown widgets).

Usage

```
block_combobox(
  input_id,
  choices,
  selected = NULL,
  placeholder = NULL,
  search_placeholder = NULL,
  empty_message = NULL,
  disabled = FALSE,
  width = NULL,
  class = NULL,
  size = c("default", "sm", "lg"),
  style = NULL,
  invalid = FALSE,
  multiple = FALSE,
  max_items = NULL
)
```

Arguments

`input_id` Input id.

choices	Choice labels and values.
selected	Optional selected value. When <code>multiple = TRUE</code> , use a character vector.
placeholder	Optional placeholder shown when no value is selected.
search_placeholder	Optional placeholder shown in the filter box. Defaults to "Search...".
empty_message	Message shown in the popup when the filter matches no choices. Defaults to "No results found.".
disabled	Whether the control is disabled.
width	Optional CSS width value.
class	Additional classes.
size	Select size. One of "default", "sm", or "lg".
style	Inline CSS styles.
invalid	Whether to show the invalid/error state.
multiple	Whether multiple values can be selected.
max_items	Optional maximum number of selected values when <code>multiple = TRUE</code> . An initial selected longer than <code>max_items</code> is an error; the runtime blocks adds beyond the cap and clamps any later server-sent selected to it.

Value

An `htmltools` tag.

See Also

`block_select()` for a non-searchable dropdown.

Other forms: `block_checkbox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

block_dark_mode_toggle

Create a dark mode toggle

Description

Create a dark mode toggle

Usage

```
block_dark_mode_toggle(label = "Theme", class = NULL)
```

Arguments

label	Button label.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other theme: [block_style\(\)](#), [block_style_profiles\(\)](#), [block_theme\(\)](#), [block_theme_presets\(\)](#), [update_block_theme\(\)](#)

block_date_picker	<i>Create a shadcn-style date picker</i>
-------------------	--

Description

A package-owned runtime input that renders a trigger button plus a popover calendar instead of wrapping Shiny's native [shiny::dateInput\(\)](#). The server value matches `dateInput()`: `input$<id>` is a length-1 Date. The control transports an ISO yyyy-mm-dd string over a `shiny.date`-typed binding, so R deserializes it as a Date with no custom handler.

Usage

```
block_date_picker(  
  input_id,  
  value = NULL,  
  min = NULL,  
  max = NULL,  
  placeholder = "Pick a date",  
  format = "yyyy-mm-dd",  
  weekstart = 0,  
  disabled = FALSE,  
  invalid = FALSE,  
  width = NULL,  
  class = NULL,  
  style = NULL  
)
```

Arguments

input_id	Input id.
value	Initial date. Accepts a Date, a POSIX time, or a "yyyy-mm-dd" string. NULL (the default) starts empty.
min	Earliest selectable date, in the same accepted forms as value. NULL for no lower bound.
max	Latest selectable date, in the same accepted forms as value. NULL for no upper bound.
placeholder	Text shown on the trigger before a date is selected.
format	Display format for the trigger label. Supports the shiny::dateInput() token set: yyyy/yy (4- or 2-digit year), mm/m (month number, zero-padded or not), MM/M (full or short month name), dd/d (day of month, zero-padded or not), and DD/D (full or short weekday name). The transported value is always ISO regardless of format.
weekstart	First day of the week, integer 0-6 using Shiny's convention (0 = Sunday, 6 = Saturday).
disabled	Whether the control is disabled.
invalid	Whether the control should show invalid styling (sets aria-invalid="true").
width	Optional CSS width value (applied to the wrapper).
class	Additional classes for the wrapper.
style	Inline CSS styles for the trigger.

Details

Unlike dateInput(), value = NULL keeps the control empty (placeholder first) rather than defaulting to today. This is intentional and matches shadcn's Date Picker examples.

Value

An htmltools tag.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

block_date_range_picker

Create a shadcn-style date range picker

Description

A package-owned runtime input that renders a trigger button plus a popover calendar for selecting a start/end date range, instead of wrapping Shiny's native `shiny::dateRangeInput()`. The server value matches `dateRangeInput()`: `input$<id>` is a length-2 Date `c(start, end)`. The control transports a two-element ISO `yyyy-mm-dd` array over a `shiny.date`-typed binding, so R deserializes it as a Date with no custom handler. An empty or incomplete range reports `NULL`.

Usage

```
block_date_range_picker(
  input_id,
  start = NULL,
  end = NULL,
  min = NULL,
  max = NULL,
  separator = " - ",
  placeholder = "Pick a date range",
  format = "yyyy-mm-dd",
  weekstart = 0,
  disabled = FALSE,
  invalid = FALSE,
  width = NULL,
  class = NULL,
  style = NULL
)
```

Arguments

<code>input_id</code>	Input id.
<code>start</code>	Initial range start. Accepts a Date, a POSIX time, or a "yyyy-mm-dd" string. <code>NULL</code> (the default) starts empty.
<code>end</code>	Initial range end, in the same accepted forms as <code>start</code> . <code>NULL</code> (the default) starts empty.
<code>min</code>	Earliest selectable date, in the same accepted forms as <code>start</code> . <code>NULL</code> for no lower bound.
<code>max</code>	Latest selectable date, in the same accepted forms as <code>start</code> . <code>NULL</code> for no upper bound.
<code>separator</code>	Text shown between the start and end dates on the trigger label. Defaults to an en dash, matching <code>shiny::dateRangeInput()</code> .
<code>placeholder</code>	Text shown on the trigger before a range is selected.

format	Display format for the trigger label. Supports the <code>shiny::dateInput()</code> token set (yyyy/yy, mm/m, MM/M, dd/d, DD/D). The transported value is always ISO regardless of format.
weekstart	First day of the week, integer 0-6 using Shiny's convention (0 = Sunday, 6 = Saturday).
disabled	Whether the control is disabled.
invalid	Whether the control should show invalid styling (sets <code>aria-invalid="true"</code>).
width	Optional CSS width value (applied to the wrapper).
class	Additional classes for the wrapper.
style	Inline CSS styles for the trigger.

Details

Unlike `dateRangeInput()`, `start = NULL` and `end = NULL` keep the control empty (placeholder first) rather than defaulting to today. This is intentional and matches `shadcn's` Date Range Picker examples. Providing only one of `start/end` is an error: there is no half-open initial state.

Value

An `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

block_dialog	Create a dialog
--------------	-----------------

Description

A modal dialog rendered into the runtime portal root. Reports its open/closed state to `input$<id>` and accepts server-driven updates through `update_block_dialog()`. The runtime implements the modal accessibility contract: Escape and overlay (outside) click dismiss, focus moves into the dialog on open and returns to the previously focused element on close, Tab/Shift+Tab cycle within the dialog, and body scroll is locked while open.

Usage

```

block_dialog(
  id,
  title,
  ...,
  description = NULL,
  footer = NULL,
  trigger = NULL,
  open = FALSE,
  size = c("default", "sm", "lg", "xl"),
  hide_title = FALSE,
  class = NULL,
  style = NULL
)

```

Arguments

<code>id</code>	Required input id. <code>input\$<id></code> is TRUE when open, FALSE when closed.
<code>title</code>	Required dialog title. Used as the accessible name.
<code>...</code>	Body content, serialized to HTML. Children are not Shiny-bound.
<code>description</code>	Optional description below the title.
<code>footer</code>	Optional footer content (typically action buttons). Renders below the body in a right-aligned flex row.
<code>trigger</code>	Optional label string. Renders a default-variant <code>block_button()</code> next to the mount node that opens the dialog when clicked. Pass NULL (default) to drive open state purely from the server with <code>update_block_dialog()</code> .
<code>open</code>	Initial open state. Defaults to FALSE.
<code>size</code>	Content max-width preset. One of "sm", "default", "lg", "xl". Defaults to "default" (32rem).
<code>hide_title</code>	Whether to visually hide the title while keeping it available to assistive technology as the dialog's accessible name. Defaults to FALSE.
<code>class</code>	Additional classes for the dialog content container.
<code>style</code>	Optional inline CSS styles for the dialog content container.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#),

```
block_table(), block_toaster(), block_tooltip(), block_value_box(), dismiss_toast(),
dropdown_menu_item(), dropdown_menu_label(), dropdown_menu_separator(), inc_block_progress(),
show_toast(), table_column(), update_block_accordion(), update_block_alert_dialog(),
update_block_dialog(), update_block_dropdown_menu(), update_block_popover(), update_block_progress(),
update_block_table(), update_block_toaster()
```

block_dropdown_menu *Create a dropdown menu*

Description

A portal-rendered action menu anchored to a trigger. Build the menu from [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), and [dropdown_menu_separator\(\)](#) parts. Choosing an item reports its value to `input$<id>` as an event (fires again even when the same item is chosen twice), so treat it like an action button that carries a value.

Usage

```
block_dropdown_menu(
  trigger,
  ...,
  id = NULL,
  label = NULL,
  side = c("bottom", "top", "left", "right"),
  align = c("start", "center", "end"),
  trigger_variant = c("outline", "default", "secondary", "ghost", "destructive", "link"),
  disabled = FALSE,
  style = NULL,
  class = NULL
)
```

Arguments

trigger	Trigger content. A single string renders a default-variant button label; an <code>htmltools</code> tag (icon button, avatar, ...) is rendered inside the trigger button as-is. Keep tag content inline and non-interactive to avoid nested buttons.
...	Menu parts created with dropdown_menu_item() , dropdown_menu_label() , and dropdown_menu_separator() .
id	Optional input id. When supplied, <code>input\$<id></code> reports the value of the most recently chosen item as an event.
label	Optional accessible name for the trigger. Recommended when trigger is an icon-only tag.
side	Side of the trigger to anchor on. One of "bottom", "top", "left", "right". Defaults to "bottom".
align	Alignment along the anchored side. One of "start", "center", "end". Defaults to "start".

trigger_variant	Button variant for a string trigger. One of the block_button() variants. Defaults to "outline".
disabled	Whether the trigger is disabled.
style	Optional inline CSS applied to the menu content container (string or named list).
class	Additional classes for the menu content container.

Details

The menu owns portal, focus, keyboard navigation (arrows, home/end, enter/space, escape, typeahead), and dismiss behavior. Focus returns to the trigger on close.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

<code>block_empty</code>	<i>Create an empty state</i>
--------------------------	------------------------------

Description

Create an empty state

Usage

```
block_empty(  
  title,  
  ...,  
  description = NULL,  
  icon = NULL,  
  action = NULL,  
  class = NULL,  
  style = NULL  
)
```

Arguments

title	Empty-state title.
...	Additional empty-state body content.
description	Optional description.
icon	Optional icon tag or vendored icon name.
action	Optional action content.
class	Additional classes.
style	Optional inline custom styles.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

block_field	<i>Create a field wrapper</i>
-------------	-------------------------------

Description

Create a field wrapper

Usage

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

Arguments

...	Field content.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

block_field_description
<i>Create field helper text</i>

Description

Create field helper text

Usage

```
block_field_description(..., id = NULL, class = NULL)
```

Arguments

...	Description content.
id	Optional element id.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

block_field_group	Create a field group
-------------------	----------------------

Description

Create a field group

Usage

block_field_group(..., class = NULL)

Arguments

- ... Field content.
- class Additional classes.

Value

An htmltools tag.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

block_field_invalid	Mark a field invalid
---------------------	----------------------

Description

Mark a field invalid

Usage

block_field_invalid(field, message)

Arguments

field	A <code>block_field()</code> tag.
message	Validation message shown below the control.

Value

A modified `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

<code>block_field_label</code>	<i>Create a field label</i>
--------------------------------	-----------------------------

Description

Create a field label

Usage

```
block_field_label(..., `for` = NULL, class = NULL)
```

Arguments

...	Label content.
for	Optional input id referenced by the label.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

block_field_legend	Create a field set legend
--------------------	---------------------------

Description

Create a field set legend

Usage

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

Arguments

...	Legend content.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

block_field_set	Create a field set
-----------------	--------------------

Description

Create a field set

Usage

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

Arguments

...	Field set content.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

block_file_input	Create a styled file input
------------------	----------------------------

Description

Runtime-rendered file picker that delegates upload transport and progress to Shiny's native file upload binding. The server receives the same `input$<id>` data frame as `shiny::fileInput()`.

Usage

```

block_file_input(
  input_id,
  variant = c("button", "dropzone"),
  multiple = FALSE,
  accept = NULL,
  button_label = "Browse",
  placeholder = "No file selected",
  dropzone_label = "Drag files here or click to browse",
  dropzone_hint = NULL,
  dropzone_icon = NULL,
  dropzone_content = NULL,
  width = NULL,
  disabled = FALSE,
  invalid = FALSE,
  style = NULL,
  class = NULL
)

```

Arguments

<code>input_id</code>	Input id.
<code>variant</code>	Picker variant. One of "button" (a styled trigger button with filename text) or "dropzone" (a focusable drag-and-drop surface). The dropzone is cosmetic chrome over the same native Shiny upload binding; <code>input\$<id></code> is identical for both variants.
<code>multiple</code>	Whether to allow selecting more than one file.
<code>accept</code>	Optional character vector of accepted MIME types or file extensions. Values are comma-joined for the native accept attribute.
<code>button_label</code>	Text shown on the picker button.
<code>placeholder</code>	Text shown before a file is selected.
<code>dropzone_label</code>	Primary text shown inside the dropzone surface (only used when <code>variant = "dropzone"</code>).
<code>dropzone_hint</code>	Secondary hint text shown beneath <code>dropzone_label</code> (only used when <code>variant = "dropzone"</code>).
<code>dropzone_icon</code>	Optional icon shown above the dropzone label (only used when <code>variant = "dropzone"</code>). Either a shinyblocks icon name (string, e.g. "upload") or an htmltools tag (e.g. an <code><svg></code>). Rendered inside a muted circle.
<code>dropzone_content</code>	Optional htmltools tag or tagList rendered as the full dropzone interior, replacing the default icon/label/hint stack (only used when <code>variant = "dropzone"</code>). Use plain htmltools markup (text, img, a styled <code><button></code>); nested <code>block_*()</code> runtime components are not hydrated inside this slot. When supplied, the surface becomes a pure drop region: mark the element that should open the file picker with <code>`data-dropzone-trigger` = NA</code> (a real <code><button>/<a></code> for keyboard support). Give it <code>class = "sb-file-dropzone-trigger"</code> for the default button styling.

width	Optional CSS width value (applied to the wrapper).
disabled	Whether the control is disabled.
invalid	Whether the control should show invalid styling (sets aria-invalid="true").
style	Inline CSS styles for the visible control.
class	Additional classes for the visible control.

Value

An `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

block_grid	<i>Create a responsive content grid</i>
------------	---

Description

Arrange repeated content in a responsive auto-fit grid whose columns shrink safely to the available width.

Usage

```
block_grid(  
  ...,  
  min_width = "16rem",  
  gap = c("md", "sm", "lg"),  
  align = c("stretch", "start", "center", "end"),  
  class = NULL  
)
```

Arguments

... Child content to arrange. Named arguments are applied to the container as HTML attributes (the `htmltools` convention), e.g. `id`, `style`, or `data-*`. The grid's responsive track (`--sb-grid-min`) is managed from the validated `min_width` and is authoritative: a caller style cannot override it.

min_width	Minimum preferred column width as a single non-negative CSS length or percentage (e.g. "16rem", 280, "50%"). calc() and CSS-wide keywords are not accepted.
gap	Spacing between children: "sm", "md", or "lg".
align	Cross-axis alignment: "stretch", "start", "center", or "end".
class	Additional classes.

Value

An htmltools tag.

See Also

Other layout: [block_body\(\)](#), [block_cluster\(\)](#), [block_header\(\)](#), [block_page\(\)](#), [block_sidebar\(\)](#), [block_stack\(\)](#)

block_header	<i>Create a dashboard header</i>
--------------	----------------------------------

Description

Create a dashboard header

Usage

block_header(..., class = NULL)

Arguments

...	Header content.
class	Additional classes.

Value

An htmltools tag.

See Also

Other layout: [block_body\(\)](#), [block_cluster\(\)](#), [block_grid\(\)](#), [block_page\(\)](#), [block_sidebar\(\)](#), [block_stack\(\)](#)

block_icon	Create an icon
------------	----------------

Description

Create an icon

Usage

```
block_icon(
  name,
  size = c("default", "sm", "lg", "xl"),
  class = NULL,
  ...,
  color = c("default", "muted", "primary", "destructive", "success", "warning", "info")
)
```

Arguments

name	Icon name from the vendored Lucide sprite, or a custom <code>htmltools</code> tag to pass through.
size	Icon size. One of "default" (1rem, the shadcn default), "sm" (0.875rem), "lg" (1.5rem), or "xl" (2.25rem). Ignored when name is a custom <code>htmltools</code> tag.
class	Additional classes.
...	Additional attributes passed to the root <code>svg</code> tag.
color	Semantic foreground color.

Value

An `htmltools` tag.

block_image_output	Frame a reactive image output
--------------------	-------------------------------

Description

Wraps `shiny::imageOutput()` in a shadcn-styled frame (aspect box, object-fit, border, radius, optional caption). App-author server code stays vanilla Shiny: `output$id <- shiny::renderImage(...)` is unchanged. The image's accessible name (`alt`) is server-controlled via `shiny::renderImage()`'s returned `alt`; the frame cannot set it.

Usage

```
block_image_output(
  id,
  width = "100%",
  height = NULL,
  aspect = NULL,
  fit = c("cover", "contain", "fill", "none", "scale-down"),
  border = FALSE,
  rounded = TRUE,
  caption = NULL,
  click = NULL,
  dblclick = NULL,
  hover = NULL,
  brush = NULL,
  inline = FALSE,
  fill = FALSE,
  class = NULL,
  style = NULL
)
```

Arguments

<code>id</code>	Shiny output id, passed verbatim to <code>shiny::imageOutput()</code> .
<code>width</code> , <code>height</code>	CSS lengths forwarded to the Shiny output. <code>height = NULL</code> resolves to "100%" when <code>aspect</code> is set, otherwise Shiny's default.
<code>aspect</code>	Aspect ratio for the media box: <code>NULL</code> , a positive number, or a "w/h" string (e.g. "16/9").
<code>fit</code>	object-fit for the rendered image. One of "cover", "contain", "fill", "none", "scale-down".
<code>border</code>	Draw a border around the media box.
<code>rounded</code>	Round the media box corners (and clip overflow).
<code>caption</code>	Optional <code><figcaption></code> text shown below the image.
<code>click</code> , <code>dblclick</code> , <code>hover</code> , <code>brush</code>	Forwarded to the Shiny output unchanged.
<code>inline</code> , <code>fill</code>	Forwarded to the Shiny output. <code>fill</code> defaults to <code>FALSE</code> to match <code>shiny::imageOutput()</code> .
<code>class</code>	Additional classes for the <code><figure></code> wrapper.
<code>style</code>	Inline style for the <code><figure></code> wrapper.

Details

For *static* images use `htmltools::img()` — no component needed. Interactive `htmlwidgets` (plotly, leaflet, DT, ...) are a different mechanism and out of scope.

Value

An `htmltools <figure>` tag.

See Also

Other outputs: [block_plot_output\(\)](#)

block_input	Create a styled single-line text input
-------------	--

Description

Create a styled single-line text input

Usage

```
block_input(  
  input_id,  
  value = "",  
  placeholder = NULL,  
  type = c("text", "password", "email", "url", "tel", "search", "number"),  
  min = NULL,  
  max = NULL,  
  step = NULL,  
  width = NULL,  
  disabled = FALSE,  
  invalid = FALSE,  
  style = NULL,  
  class = NULL  
)
```

Arguments

input_id	Input id.
value	Initial value.
placeholder	Optional placeholder text.
type	Input type. One of "text", "password", "email", "url", "tel", "search", or "number". Defaults to "text".
min	Optional numeric lower bound. Only valid when type = "number".
max	Optional numeric upper bound. Only valid when type = "number".
step	Optional positive step size for the stepper buttons and arrow keys. Only valid when type = "number"; defaults to 1 in the browser when unset.
width	Optional CSS width value (applied to the wrapper).
disabled	Whether the control is disabled.
invalid	Whether the control should show invalid styling (sets aria-invalid="true").
style	Inline CSS styles for the input element.
class	Additional classes for the wrapper.

Details

When `type = "number"`, the control renders increment/decrement stepper buttons and `input$<input_id>` reports a numeric value (like `shiny::numericInput()`): NA while the field is empty or unparseable, a numeric scalar otherwise. All other types report a character string. The reported type is fixed when the control first binds; switching to or from "number" with `update_block_input()` does not change it.

Value

An `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

block_input_group	Create an input group
-------------------	-----------------------

Description

Create an input group

Usage

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

Arguments

<code>...</code>	Input group content.
<code>class</code>	Additional classes.

Value

An `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

`block_input_group_addon`

Create an input group addon

Description

Create an input group addon

Usage

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

Arguments

<code>...</code>	Addon content.
<code>class</code>	Additional classes.

Value

An `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

block_nav	Create a navigation container
-----------	-------------------------------

Description

When `id` is supplied the navigation becomes a Shiny input: clicking a `block_nav_item()` reports that item's value as `input[[id]]` and moves the selected highlight, so the sidebar can drive page navigation the same way a Shiny tabset does. Without `id` the items stay plain links.

Usage

```
block_nav(..., id = NULL, class = NULL)
```

Arguments

<code>...</code>	Navigation items.
<code>id</code>	Optional Shiny input id. When set, the selected item's value is reported as <code>input[[id]]</code> ; pair it with <code>shiny::conditionalPanel()</code> or <code>shiny::renderUI()</code> to switch pages, and <code>update_block_nav()</code> to select from the server.
<code>class</code>	Additional classes.

Value

An `htmltools` tag.

See Also

Other navigation: `block_breadcrumb()`, `block_breadcrumb_ellipsis()`, `block_breadcrumb_item()`, `block_nav_group()`, `block_nav_item()`, `block_nav_label()`, `block_tab()`, `block_tabs()`, `update_block_nav()`, `update_block_tabs()`

block_nav_group	Create a collapsible sidebar navigation group
-----------------	---

Description

Groups wrap leaf `block_nav_item()` children in a disclosure region. The group trigger toggles expansion only; it never reports a Shiny input value.

Usage

```
block_nav_group(  
  label,  
  ...,  
  icon = NULL,  
  value = NULL,  
  expanded = TRUE,  
  class = NULL  
)
```

Arguments

label	Group label.
...	Child block_nav_item() tags. Named arguments are applied to the group container as HTML attributes.
icon	Optional icon tag or vendored icon name.
value	Optional group identity for expansion state hooks. Stored as data-sb-nav-group-value; it is not reported as a nav input value.
expanded	Whether the group starts expanded.
class	Additional classes for the group container.

Value

An `htmltools` tag.

See Also

Other navigation: [block_breadcrumb\(\)](#), [block_breadcrumb_ellipsis\(\)](#), [block_breadcrumb_item\(\)](#), [block_nav\(\)](#), [block_nav_item\(\)](#), [block_nav_label\(\)](#), [block_tab\(\)](#), [block_tabs\(\)](#), [update_block_nav\(\)](#), [update_block_tabs\(\)](#)

block_nav_item	<i>Create a sidebar navigation item</i>
----------------	---

Description

Create a sidebar navigation item

Usage

```
block_nav_item(  
  label,  
  value = NULL,  
  href = "#",  
  icon = NULL,  
  selected = FALSE,  
  class = NULL  
)
```

Arguments

label	Navigation label.
value	Value reported as the input when the parent <code>block_nav()</code> has an id. Defaults to label when it is a single string.
href	Destination URL. Ignored when the parent <code>block_nav()</code> is an input (the click selects the item instead of following the link).
icon	Optional icon tag or vendored icon name.
selected	Whether the item is selected.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other navigation: `block_breadcrumb()`, `block_breadcrumb_ellipsis()`, `block_breadcrumb_item()`, `block_nav()`, `block_nav_group()`, `block_nav_label()`, `block_tab()`, `block_tabs()`, `update_block_nav()`, `update_block_tabs()`

block_nav_label	<i>Create a sidebar navigation section label</i>
-----------------	--

Description

Section labels are non-interactive captions inside `block_nav()`. They do not report a Shiny input value.

Usage

```
block_nav_label(text, class = NULL)
```

Arguments

text	Label text.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other navigation: `block_breadcrumb()`, `block_breadcrumb_ellipsis()`, `block_breadcrumb_item()`, `block_nav()`, `block_nav_group()`, `block_nav_item()`, `block_tab()`, `block_tabs()`, `update_block_nav()`, `update_block_tabs()`

block_page

Create a modern Shiny page

Description

Create a modern Shiny page

Usage

```
block_page(
  ...,
  title = NULL,
  sidebar = NULL,
  header = NULL,
  theme_mode = c("system", "light", "dark"),
  theme = NULL,
  style = NULL,
  class = NULL
)
```

Arguments

...	Page body content.
title	Browser page title.
sidebar	Optional sidebar content.
header	Optional header content.
theme_mode	Initial theme mode.
theme	Optional <code>block_theme()</code> overrides.
style	Optional <code>block_style()</code> visual style profile. When supplied, <code>data-sb-style="<profile>"</code> is placed on the <code>.sb-app</code> shell and any profile override <code><style></code> is injected, so the profile applies page-wide (including portal overlays).
class	Additional classes for the app root.

Value

An `htmltools` tag list suitable for a Shiny UI.

See Also

Other layout: `block_body()`, `block_cluster()`, `block_grid()`, `block_header()`, `block_sidebar()`, `block_stack()`

block_pagination	Create a pagination input
------------------	---------------------------

Description

Renders a shadcn-style pagination control with previous, next, numbered, and ellipsis controls. The selected page is reported as an integer through `input$<input_id>`.

Usage

```
block_pagination(
  input_id,
  pages,
  selected = 1,
  sibling_count = 1,
  show_edges = TRUE,
  disabled = FALSE,
  style = NULL,
  class = NULL
)
```

Arguments

<code>input_id</code>	Input id.
<code>pages</code>	Total number of pages; a positive whole number.
<code>selected</code>	Initially selected page.
<code>sibling_count</code>	Number of page links shown on each side of the selected page.
<code>show_edges</code>	Whether to keep the first and last page visible.
<code>disabled</code>	Whether the entire control is disabled.
<code>style</code>	Inline CSS styles applied to the pagination wrapper.
<code>class</code>	Additional classes for the wrapper.

Value

An `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

block_plot_output	<i>Frame a reactive plot output</i>
-------------------	-------------------------------------

Description

Wraps `shiny::plotOutput()` in a shadcn-styled frame (aspect box, border, radius, optional caption). App-author server code stays vanilla Shiny: `output$id <- shiny::renderPlot(...)` is unchanged. Covers base graphics, ggplot2, and lattice. The plot's accessible name (`alt`) is server-controlled via `shiny::renderPlot()`'s `alt`; the frame cannot set it.

Usage

```
block_plot_output(
  id,
  width = "100%",
  height = NULL,
  aspect = NULL,
  border = FALSE,
  rounded = TRUE,
  caption = NULL,
  click = NULL,
  dblclick = NULL,
  hover = NULL,
  brush = NULL,
  inline = FALSE,
  fill = !inline,
  class = NULL,
  style = NULL
)
```

Arguments

<code>id</code>	Shiny output id, passed verbatim to <code>shiny::plotOutput()</code> .
<code>width, height</code>	CSS lengths forwarded to the Shiny output. <code>height = NULL</code> resolves to "100%" when <code>aspect</code> is set, otherwise Shiny's default.
<code>aspect</code>	Aspect ratio for the media box: <code>NULL</code> , a positive number, or a "w/h" string (e.g. "16/9").
<code>border</code>	Draw a border around the media box.
<code>rounded</code>	Round the media box corners (and clip overflow).
<code>caption</code>	Optional <code><figcaption></code> text shown below the plot.
<code>click, dblclick, hover, brush</code>	Forwarded to the Shiny output unchanged.
<code>inline, fill</code>	Forwarded to the Shiny output. <code>fill</code> defaults to <code>!inline</code> to match <code>shiny::plotOutput()</code> .
<code>class</code>	Additional classes for the <code><figure></code> wrapper.
<code>style</code>	Inline style for the <code><figure></code> wrapper.

Details

Unlike `block_image_output()` there is no `fit` argument: `shiny::renderPlot()` already renders to the media box size, so `object-fit` would have no visible effect.

Value

An `htmltools <figure>` tag.

See Also

Other outputs: `block_image_output()`

block_popover	<i>Create a runtime popover</i>
---------------	---------------------------------

Description

A non-modal, portal-rendered popover anchored to a trigger button. Popovers with an `id` report their open/closed state to `input$<id>` and accept server-driven updates through `update_block_popover()`. Popovers without an `id` stay client-only.

Usage

```
block_popover(
  trigger,
  ...,
  id = NULL,
  side = c("bottom", "top", "left", "right"),
  align = c("center", "start", "end"),
  open = FALSE,
  style = NULL,
  class = NULL
)
```

Arguments

<code>trigger</code>	Required label string. Renders a default-variant <code>block_button()</code> that toggles the popover when clicked.
<code>...</code>	Popover body content. Serialized to HTML.
<code>id</code>	Optional input id. When supplied, <code>input\$<id></code> is <code>TRUE</code> when open and <code>FALSE</code> when closed.
<code>side</code>	Side of the trigger to anchor on. One of <code>"bottom"</code> , <code>"top"</code> , <code>"left"</code> , <code>"right"</code> . Defaults to <code>"bottom"</code> .
<code>align</code>	Alignment along the anchored side. One of <code>"center"</code> , <code>"start"</code> , <code>"end"</code> . Defaults to <code>"center"</code> .

open	Initial open state. Defaults to FALSE.
style	Optional inline CSS applied to the popover content container (string or named list).
class	Additional classes for the popover content container.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

block_progress	<i>Create a progress bar</i>
----------------	------------------------------

Description

An embedded, shadcn-style progress indicator. Unlike Shiny's native [shiny::Progress](#) notification panel, this renders inline exactly where it is placed in the UI. It is display-only: it exposes no meaningful input `id` value, but the server can drive it with [update_block_progress\(\)](#) and [inc_block_progress\(\)](#).

Usage

```
block_progress(
  id,
  value = 0,
  min = 0,
  max = 1,
  message = NULL,
  detail = NULL,
  label = NULL,
  show_value = FALSE,
  indeterminate = FALSE,
  variant = c("default", "success", "warning", "info", "destructive"),
  width = NULL,
  class = NULL,
```

```

    style = NULL
)

```

Arguments

id	Component id, used to address the bar from the server. Not a form control: there is no input\$<id> value.
value	Current progress value. Clamped into [min, max].
min	Lower bound. Must be finite and less than max.
max	Upper bound. Must be finite and greater than min.
message	Dynamic status line (e.g. "Importing rows..."). Renders at header-left, or as a muted second line when label is also set.
detail	Secondary muted text below the track.
label	Static description of what is progressing (e.g. "Upload"). Takes header-left when set.
show_value	Whether to render the clamped percent at header-right. Suppressed in indeterminate mode.
indeterminate	Whether the bar shows an unknown-progress sweep instead of a determinate fill.
variant	Indicator color: one of "default", "success", "warning", "info", "destructive".
width	Optional CSS width for the component (NULL fills the container). Sizes the mount wrapper only.
class	Additional classes applied to the bar element (.sb-progress-body), not the mount wrapper.
style	Optional inline styles (CSS string or named list) applied to the bar element. Targets the same node as update_block_progress(style =); use width to size the component.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

block_radio_group	Create a styled radio group input
-------------------	-----------------------------------

Description

Renders a shadcn-style radio group where exactly one option is selected at a time. Reports the selected value through a package-local Shiny input binding.

Usage

```
block_radio_group(
  input_id,
  choices,
  selected = NULL,
  disabled = FALSE,
  invalid = FALSE,
  orientation = c("vertical", "horizontal"),
  style = NULL,
  class = NULL
)
```

Arguments

input_id	Input id.
choices	Choice labels and values. A named character vector (c(Label = "value")), a list, or a character vector.
selected	Optional initial value. Must match one of choices.
disabled	Whether the entire group is disabled.
invalid	Whether the group should show invalid styling (sets aria-invalid="true" on the wrapper).
orientation	Either "vertical" (default) or "horizontal".
style	Inline CSS styles applied to the radio-group wrapper.
class	Additional classes for the wrapper.

Value

An `htmltools` tag.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#),

```
update_block_combobox(), update_block_date_picker(), update_block_date_range_picker(),
update_block_file_input(), update_block_input(), update_block_pagination(), update_block_radio_group(),
update_block_select(), update_block_slider(), update_block_switch(), update_block_textarea(),
update_block_toggle_group()
```

block_select

Create a styled select input

Description

block_select() renders a hidden native <select> as the Shiny input value source, plus a package runtime overlay for the visible shadcn-style trigger and popup.

Usage

```
block_select(
  input_id,
  choices,
  selected = NULL,
  placeholder = NULL,
  disabled = FALSE,
  width = NULL,
  class = NULL,
  size = c("default", "sm", "lg"),
  style = NULL,
  invalid = FALSE,
  multiple = FALSE,
  max_items = NULL
)
```

Arguments

input_id	Input id.
choices	Choice labels and values.
selected	Optional selected value. When multiple = TRUE, use a character vector.
placeholder	Optional placeholder shown when no value is selected.
disabled	Whether the control is disabled.
width	Optional CSS width value.
class	Additional classes.
size	Select size. One of "default", "sm", or "lg".
style	Inline CSS styles.
invalid	Whether to show the invalid/error state.
multiple	Whether multiple values can be selected.
max_items	Optional maximum number of selected values when multiple = TRUE. An initial selected longer than max_items is an error; the runtime blocks adds beyond the cap and clamps any later server-sent selected to it.

Value

An `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

<code>block_separator</code>	<i>Create a separator</i>
------------------------------	---------------------------

Description

Create a separator

Usage

```
block_separator(  
  orientation = c("horizontal", "vertical"),  
  decorative = TRUE,  
  class = NULL  
)
```

Arguments

- `orientation` Separator orientation.
- `decorative` Whether the separator is decorative only.
- `class` Additional classes.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

block_sidebar	Create a dashboard sidebar
---------------	----------------------------

Description

Create a dashboard sidebar

Usage

```
block_sidebar(
  ...,
  title = NULL,
  collapsible = FALSE,
  collapsed = FALSE,
  id = NULL,
  class = NULL
)
```

Arguments

...	Sidebar content.
title	Optional sidebar title.
collapsible	Whether the sidebar can collapse on larger screens.
collapsed	Whether the sidebar starts collapsed on larger screens.
id	Optional sidebar DOM id.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other layout: [block_body\(\)](#), [block_cluster\(\)](#), [block_grid\(\)](#), [block_header\(\)](#), [block_page\(\)](#), [block_stack\(\)](#)

block_skeleton	Create a skeleton placeholder
----------------	-------------------------------

Description

Create a skeleton placeholder

Usage

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

Arguments

class	Additional classes.
...	Additional attributes passed to <code>htmltools::tags\$div</code> .

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

block_slider	Create a styled slider input
--------------	------------------------------

Description

Runtime-rendered slider with a dedicated Shiny input binding. Supports single-value and two-value range sliders.

Usage

```
block_slider(  
  input_id,  
  value,  
  min,  
  max,  
  step = NULL,  
  ticks = FALSE,  
  orientation = c("horizontal", "vertical"),  
  show_value = FALSE,  
  min_label = NULL,  
  max_label = NULL,  
  width = NULL,  
  disabled = FALSE,  
  invalid = FALSE,  
  style = NULL,  
  class = NULL  
)
```

Arguments

<code>input_id</code>	Input id.
<code>value</code>	Initial value. Length 1 for a single-handle slider; length 2 for a range slider.
<code>min</code>	Numeric lower bound.
<code>max</code>	Numeric upper bound.
<code>step</code>	Step size. Defaults to NULL (Shiny's auto-step).
<code>ticks</code>	Whether to show tick marks on the rail.
<code>orientation</code>	Slider orientation. One of "horizontal" or "vertical".
<code>show_value</code>	Whether to render the current value above the slider.
<code>min_label</code>	Optional label displayed at the minimum end of the rail.
<code>max_label</code>	Optional label displayed at the maximum end of the rail.
<code>width</code>	Optional CSS width value for horizontal sliders.
<code>disabled</code>	Whether the control is disabled.
<code>invalid</code>	Whether the control should show invalid styling (sets <code>aria-invalid="true"</code>).
<code>style</code>	Inline CSS styles for the slider control.
<code>class</code>	Additional classes for the wrapper.

Value

An `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

block_spinner	Create a spinner
---------------	------------------

Description

Create a spinner

Usage

```
block_spinner(  
  label = "Loading",  
  size = c("default", "sm", "lg"),  
  color = c("default", "muted", "primary", "destructive", "success", "warning", "info"),  
  icon = spinner_icon_choices(),  
  class = NULL,  
  style = NULL  
)
```

Arguments

label	Accessible aria-label announced by assistive technology.
size	Visual size.
color	Semantic color.
icon	Spinner icon name. One of "loader-2", "loader", "loader-circle", "loader-pinwheel", "refresh-cw", "rotate-cw", or "circle-dashed".
class	Additional classes.
style	Optional inline custom styles.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

block_stack	<i>Stack content vertically</i>
-------------	---------------------------------

Description

Arrange content in a vertical flow with package-owned semantic spacing.

Usage

```
block_stack(  
  ...,  
  gap = c("md", "sm", "lg"),  
  align = c("stretch", "start", "center", "end"),  
  class = NULL  
)
```

Arguments

- ... Child content to arrange. Named arguments are applied to the container as HTML attributes (the `htmltools` convention), e.g. `id`, `style`, or `data-*`. Layout itself (display, direction, gap, alignment) is owned by the primitive's classes — use `gap/align`, not an inline style, to control it.
- gap Spacing between children: "sm", "md", or "lg".
- align Cross-axis alignment: "stretch", "start", "center", or "end".
- class Additional classes.

Value

An `htmltools` tag.

See Also

Other layout: [block_body\(\)](#), [block_cluster\(\)](#), [block_grid\(\)](#), [block_header\(\)](#), [block_page\(\)](#), [block_sidebar\(\)](#)

block_style

Select a visual style profile

Description

Selects a built-in visual style profile and, optionally, layers explicit overrides on top of it. A *style profile* controls visual feel — control sizing, spacing, surface and overlay metrics, elevation, focus and disabled treatment, and motion — through a curated set of `--sb-*` custom properties. This is separate from `block_theme()`, which owns semantic light/dark **colour** tokens. Portaled overlays stay inside their originating runtime root and therefore retain scoped profile overrides.

Usage

```
block_style(profile = "default", ..., scope = NULL)
```

Arguments

profile	Built-in style profile name. Defaults to "default", which preserves the current shinyblocks visuals.
...	Named token overrides from the curated allowlist, such as <code>control_height = "2.5rem"</code> . Override values win over the profile's built-in values.
scope	Optional CSS selector. When supplied, the profile's tokens (and any ... overrides) apply only to elements matching scope (and the runtime/portal roots inside it) instead of the whole page. Defaults to NULL (page-wide). Scope covers every token-driven part of a profile — radii, surfaces, borders, shadows, and the shared <code>--sb-*</code> tokens. A profile may also carry a little genuinely-structural CSS (for example a switch's enlarged geometry or a dialog's blurred scrim) that is keyed off a page-level <code>data-sb-style="<profile>"</code> attribute, which scope alone does not set. For full per-subtree fidelity, also place <code>data-sb-style="<profile>"</code> on the scope element (the showcase's scoped previews do this). Passing style to <code>block_page()</code> applies the profile page-wide and needs none of this.

Details

Pass the result to `block_page()` via its style argument. The profile is applied page-wide: `block_page()` places `data-sb-style="<profile>"` on the `.sb-app` shell, so profile tokens also reach dialog, popover, tooltip, and select portal content (the portal root lives inside `.sb-app`).

Overrides use ergonomic snake-case names from a fixed allowlist (for example `control_height`, `surface_padding`, `focus_ring_width`); see `block_style_profiles()` for available profiles. Raw `--sb-*` CSS-variable names are intentionally not accepted.

Value

A shinyblocks_style object consumed by `block_page()`.

See Also

Other theme: [block_dark_mode_toggle\(\)](#), [block_style_profiles\(\)](#), [block_theme\(\)](#), [block_theme_presets\(\)](#), [update_block_theme\(\)](#)

Examples

```
block_style("default")
block_style("default", control_height = "2.5rem", surface_gap = "2rem")
```

block_style_profiles	<i>Supported built-in style profiles</i>
----------------------	--

Description

Supported built-in style profiles

Usage

```
block_style_profiles()
```

Value

A character vector of built-in style-profile names accepted by [block_style\(\)](#).

See Also

Other theme: [block_dark_mode_toggle\(\)](#), [block_style\(\)](#), [block_theme\(\)](#), [block_theme_presets\(\)](#), [update_block_theme\(\)](#)

Examples

```
block_style_profiles()
```

block_switch	<i>Create a styled switch input</i>
--------------	-------------------------------------

Description

Create a styled switch input

Usage

```
block_switch(  
  input_id,  
  label,  
  value = FALSE,  
  disabled = FALSE,  
  size = c("default", "sm", "lg"),  
  style = NULL,  
  class = NULL  
)
```

Arguments

input_id	Input id.
label	Switch label.
value	Whether the switch starts on.
disabled	Whether the control is disabled.
size	Switch size. One of "default", "sm", or "lg".
style	Inline CSS styles.
class	Additional classes.

Value

An `htmltools` tag.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

block_tab	Create a tab
-----------	--------------

Description

Create a tab

Usage

```
block_tab(title, ..., value = NULL)
```

Arguments

title	Tab label.
...	Tab content.
value	Optional tab value.

Value

An htmltools tag.

See Also

Other navigation: [block_breadcrumb\(\)](#), [block_breadcrumb_ellipsis\(\)](#), [block_breadcrumb_item\(\)](#), [block_nav\(\)](#), [block_nav_group\(\)](#), [block_nav_item\(\)](#), [block_nav_label\(\)](#), [block_tabs\(\)](#), [update_block_nav\(\)](#), [update_block_tabs\(\)](#)

block_table

Render a data frame as a styled table

Description

`block_table()` renders a data frame or tibble as a runtime-owned shadcn table. Cell formatting happens in R before the payload is sent to the browser. The same formatting pipeline is reused by [update_block_table\(\)](#), so a server-side refresh produces an identical payload.

Usage

```
block_table(
  data,
  columns = NULL,
  caption = NULL,
  max_rows = NULL,
  na = "",
  digits = NULL,
  rownames = FALSE,
  row_format = NULL,
  striped = FALSE,
  hover = TRUE,
  bordered = FALSE,
  selection = c("none", "single", "multiple"),
  selected = NULL,
  id = NULL,
  class = NULL,
  style = NULL
)
```

Arguments

data	A data frame or tibble.
columns	Optional named list of per-column overrides created with <code>table_column()</code> . Names must match columns in data.
caption	Optional caption rendered below the table.
max_rows	Optional non-negative integer limiting the number of rendered rows. When rows are truncated, a footer note reports the total row count.
na	String used to render missing values. Defaults to "" (empty cell). Per-column overrides win via <code>table_column(na =)</code> .
digits	Optional non-negative integer giving the number of decimal places for default numeric formatting. NULL keeps R's default <code>format()</code> . Ignored for columns with a custom format function. Per-column overrides win via <code>table_column(digits =)</code> .
rownames	Whether to render <code>row.names(data)</code> as a leading column.
row_format	Optional function(<code>row</code> , <code>i</code>) called once per rendered row, where <code>row</code> is the original (unformatted) row as a named list and <code>i</code> is the row index. Return NULL for no styling, or a list with optional <code>intent</code> , <code>emphasis</code> , <code>class</code> , and/or <code>style</code> entries applied to that row's <code><tr></code> .
striped	Whether to zebra-stripe body rows.
hover	Whether rows highlight on hover. Defaults to TRUE (shdcn base).
bordered	Whether to draw cell borders.
selection	Row-selection mode, following the DT idiom. One of "none" (default; the table is presentational and reports no value), "single" (one row selectable at a time), or "multiple" (any number of rows). When enabled the table reports its selection to the server (see <i>Selection</i>).
selected	Optional integer vector of 1-based row indices to select on load. Only valid when selection is "single" or "multiple".
id	Optional input id. Required if the table is updated from the server with <code>update_block_table()</code> or uses row selection; static, non-selectable tables can omit it.
class	Additional classes on the runtime mount.
style	Optional inline custom styles.

Value

An `htmltools` tag.

Selection

With selection set to "single" or "multiple", rows become clickable and the table reports its selection through Shiny inputs, mirroring the DT package so existing DT code ports over:

- `input$<id>` and `input$<id>_rows_selected` – integer vector of the selected 1-based row indices (the bare `id` is a shinyblocks convenience; `_rows_selected` is the DT-compatible name).

- `input$<id>_row_last_clicked` – the 1-based index of the most recently clicked row.
- `input$<id>_cell_clicked` – a list with row (1-based), col (1-based rendered column index), and value (the displayed cell text) for the most recent click.

Push a selection from the server with `update_block_table(selected = )`.

See Also

Other content: `block_accordion()`, `block_accordion_item()`, `block_alert()`, `block_alert_action()`, `block_alert_description()`, `block_alert_dialog()`, `block_alert_title()`, `block_badge()`, `block_card()`, `block_card_content()`, `block_card_description()`, `block_card_footer()`, `block_card_header()`, `block_card_title()`, `block_code()`, `block_dialog()`, `block_dropdown_menu()`, `block_empty()`, `block_popover()`, `block_progress()`, `block_separator()`, `block_skeleton()`, `block_spinner()`, `block_toaster()`, `block_tooltip()`, `block_value_box()`, `dismiss_toast()`, `dropdown_menu_item()`, `dropdown_menu_label()`, `dropdown_menu_separator()`, `inc_block_progress()`, `show_toast()`, `table_column()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`, `update_block_dropdown_menu()`, `update_block_popover()`, `update_block_progress()`, `update_block_table()`, `update_block_toaster()`

block_tabs

Create styled tabs

Description

Create styled tabs

Usage

```
block_tabs(
  ...,
  id = NULL,
  selected = NULL,
  variant = c("default", "line"),
  orientation = c("horizontal", "vertical"),
  class = NULL
)
```

Arguments

<code>...</code>	<code>block_tab()</code> or <code>shiny::tabPanel()</code> items.
<code>id</code>	Optional input id.
<code>selected</code>	Optional selected tab value.
<code>variant</code>	Visual variant: default or line.
<code>orientation</code>	Layout orientation: horizontal or vertical.
<code>class</code>	Additional classes.

Value

An `htmltools` tag.

See Also

Other navigation: [block_breadcrumb\(\)](#), [block_breadcrumb_ellipsis\(\)](#), [block_breadcrumb_item\(\)](#), [block_nav\(\)](#), [block_nav_group\(\)](#), [block_nav_item\(\)](#), [block_nav_label\(\)](#), [block_tab\(\)](#), [update_block_nav\(\)](#), [update_block_tabs\(\)](#)

block_task_button	Create a task button with automatic busy state
-------------------	--

Description

An action button that locks itself the instant it is clicked, shows a busy label and spinner while work runs, and reports a `shinyActionButtonValue` (usable exactly like [shiny::actionButton\(\)](#)). By default it returns to the ready state after the reactive flush that the click triggered; pass `auto_reset = FALSE` to keep it busy until you release it with [update_block_task_button\(\)](#).

Usage

```
block_task_button(
  input_id,
  label,
  label_busy = "Processing. . . ",
  variant = TASK_BUTTON_VARIANTS,
  size = TASK_BUTTON_SIZES,
  icon = NULL,
  icon_busy = NULL,
  icon_position = TASK_BUTTON_ICON_POSITIONS,
  auto_reset = TRUE,
  ...,
  class = NULL
)
```

Arguments

<code>input_id</code>	Input id. Required, because the busy/ready behavior depends on a Shiny input binding. Read the click count with <code>input[[input_id]]</code> .
<code>label</code>	Button label (ready state).
<code>label_busy</code>	Accessible and visible label shown while busy.
<code>variant</code>	Visual variant.
<code>size</code>	Button size: one of "default", "sm", or "lg".
<code>icon</code>	Optional ready-state icon: a vendored icon name or shiny tag.

icon_busy	Optional busy-state icon: a vendored icon name or shiny.tag. Defaults to a spinner when NULL.
icon_position	Whether the icon appears before or after the label.
auto_reset	When TRUE (default), the button returns to ready after the reactive flush triggered by the click, unless the server has taken manual control via update_block_task_button() .
...	Additional attributes passed to the button. Pass disabled = TRUE to render disabled.
class	Additional classes merged onto the runtime button element.

Details

Inspired by shadcn's Button visuals and bslib's `input_task_button()` behavior.

Value

An `htmltools` tag.

See Also

[update_block_task_button\(\)](#)

Other action: [block_button\(\)](#), [update_block_button\(\)](#), [update_block_task_button\(\)](#)

block_textarea	<i>Create a styled textarea input</i>
----------------	---------------------------------------

Description

Create a styled textarea input

Usage

```
block_textarea(
  input_id,
  value = "",
  placeholder = NULL,
  rows = 3,
  width = NULL,
  disabled = FALSE,
  invalid = FALSE,
  resize = c("vertical", "none", "both", "horizontal"),
  style = NULL,
  class = NULL
)
```

Arguments

input_id	Input id.
value	Initial value.
placeholder	Optional placeholder text.
rows	Number of visible rows.
width	Optional CSS width value (applied to the wrapper).
disabled	Whether the control is disabled.
invalid	Whether the control should show invalid styling (sets aria-invalid="true").
resize	Textarea resize behavior. One of "vertical", "none", "both", or "horizontal".
style	Inline CSS styles for the textarea element.
class	Additional classes for the wrapper.

Value

An `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

block_theme	Create theme overrides
-------------	------------------------

Description

Emits a scoped `<style>` block that selects and overrides `shadcn` token variables. By default the overrides apply to every `.sb-app`, standalone shinyblocks scope, and runtime root. Pass `scope` to confine the overrides to a single subtree, which is essential when several differently-themed regions share one page (for example a component gallery) so a local override does not leak into the rest of the app. Portaled overlays are mounted inside their originating runtime root, so they retain the same scoped overrides.

Usage

```
block_theme(..., preset = NULL, scope = NULL, dark = NULL)
```

Arguments

<code>...</code>	Named CSS token overrides, such as <code>primary</code> , <code>background</code> , or <code>radius</code> . Applied in both light and dark mode.
<code>preset</code>	Optional built-in semantic color palette. One of <code>neutral</code> , <code>stone</code> , <code>zinc</code> , <code>mauve</code> , <code>olive</code> , <code>mist</code> , or <code>taupe</code> .
<code>scope</code>	Optional CSS selector. When supplied, overrides apply only to elements matching <code>scope</code> and the runtime roots inside it, instead of the whole page. Defaults to <code>NULL</code> (page-wide).
<code>dark</code>	Optional named list of token overrides applied only in dark mode, overriding the corresponding <code>...</code> value (or the package default) when <code>[data-theme="dark"]</code> is active. Defaults to <code>NULL</code> .

Details

Like `shadcn`, tokens have separate light and dark values. The `...` overrides apply to **both** light and dark mode. Pass `dark` to set values that apply **only** in dark mode (when `[data-theme="dark"]` is active), mirroring `shadcn`'s `:root / .dark` token pairs. A token present only in dark keeps the package default in light mode and the dark value in dark mode.

Value

An `htmltools` tag.

See Also

Other theme: [block_dark_mode_toggle\(\)](#), [block_style\(\)](#), [block_style_profiles\(\)](#), [block_theme_presets\(\)](#), [update_block_theme\(\)](#)

block_theme_presets	<i>Supported built-in colour presets</i>
---------------------	--

Description

Supported built-in colour presets

Usage

```
block_theme_presets()
```

Value

A character vector of built-in palette names accepted by the `preset` argument of [block_theme\(\)](#).

See Also

Other theme: [block_dark_mode_toggle\(\)](#), [block_style\(\)](#), [block_style_profiles\(\)](#), [block_theme\(\)](#), [update_block_theme\(\)](#)

Examples

```
block_theme_presets()
```

```
block_toaster
```

```
    Create a toast notification region
```

Description

Mounts a single, portal-rendered region that displays transient toast notifications fired from the server with `show_toast()`. Unlike `block_alert()`, a toaster is not inline content: place one (per position) near the top of your app body, then call `show_toast()` to push messages.

Usage

```
block_toaster(
  id,
  position = c("bottom-right", "bottom-center", "bottom-left", "top-right", "top-center",
    "top-left"),
  class = NULL,
  style = NULL
)
```

Arguments

<code>id</code>	Required input id. Targets this toaster from <code>show_toast()</code> and <code>dismiss_toast()</code> , and reports toast lifecycle events to <code>input\$<id></code> .
<code>position</code>	Screen corner/edge the stack anchors to. One of "top-left", "top-center", "top-right", "bottom-left", "bottom-center", "bottom-right". Defaults to "bottom-right".
<code>class</code>	Additional classes for the toaster region.
<code>style</code>	Optional inline custom styles for the toaster region.

Details

`input$<id>` reports the most recent toast lifecycle event as a list with `action` ("show" or "dismiss"), `id` (the toast id, or NULL when all toasts are dismissed), and `seq` (a monotonic counter). The counter guarantees the value changes on every show and dismiss — including auto-dismiss, the close button, and Escape — so server observers always fire.

Value

An `htmltools` tag.

See Also

`show_toast()`, `dismiss_toast()`

Other content: `block_accordion()`, `block_accordion_item()`, `block_alert()`, `block_alert_action()`, `block_alert_description()`, `block_alert_dialog()`, `block_alert_title()`, `block_badge()`, `block_card()`, `block_card_content()`, `block_card_description()`, `block_card_footer()`, `block_card_header()`, `block_card_title()`, `block_code()`, `block_dialog()`, `block_dropdown_menu()`, `block_empty()`, `block_popover()`, `block_progress()`, `block_separator()`, `block_skeleton()`, `block_spinner()`, `block_table()`, `block_tooltip()`, `block_value_box()`, `dismiss_toast()`, `dropdown_menu_item()`, `dropdown_menu_label()`, `dropdown_menu_separator()`, `inc_block_progress()`, `show_toast()`, `table_column()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`, `update_block_dropdown_menu()`, `update_block_popover()`, `update_block_progress()`, `update_block_table()`, `update_block_toaster()`

block_toggle_group	Create a segmented toggle group input
--------------------	---------------------------------------

Description

Renders a shadcn-style toggle group: a joined row of pressable buttons for single or multiple choice (view switchers, formatting toolbars). Reports the pressed value(s) through a package-local Shiny input binding: a string (or NULL when nothing is pressed) for type = "single", a character vector for type = "multiple".

Usage

```
block_toggle_group(
  input_id,
  choices,
  selected = NULL,
  type = c("single", "multiple"),
  variant = c("default", "outline"),
  size = c("default", "sm", "lg"),
  icons = NULL,
  icon_only = FALSE,
  disabled = FALSE,
  style = NULL,
  class = NULL
)
```

Arguments

<code>input_id</code>	Input id.
<code>choices</code>	Choice labels and values. A named character vector (<code>c(Label = "value")</code>), a list, or a character vector.
<code>selected</code>	Optional initial value(s). For type = "single" a single value or NULL (nothing pressed); for type = "multiple" a character vector. Must match choices.

type	"single" (radio-like, pressing another item releases the current one, pressing the active item releases it) or "multiple" (independent pressed states). Create-only.
variant	Visual variant: "default" (borderless) or "outline".
size	Item size. One of "default", "sm", or "lg".
icons	Optional named list mapping choice values to icons. Each element is a vendored icon name (see <code>block_icon()</code>) or a shiny.tag.
icon_only	Whether to hide item labels visually. Labels remain as the accessible name (aria-label), so every choice still needs a label. Requires icons covering every choice.
disabled	TRUE/FALSE to disable the whole group, or a character vector of choice values to disable individual items.
style	Inline CSS styles applied to the toggle-group wrapper.
class	Additional classes for the wrapper.

Value

An `htmltools` tag.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

block_tooltip	Create a tooltip
---------------	------------------

Description

Wraps a trigger label with a small floating panel that opens on hover or keyboard focus and closes on leave, blur, or Escape. The content panel renders through `[data-shinyblocks-portal-root]` to avoid clipping by ancestor overflow or transform. Tooltips have no Shiny input binding; treat them as purely presentational.

Usage

```

block_tooltip(
    trigger,
    ...,
    side = c("top", "bottom", "left", "right"),
    align = c("center", "start", "end"),
    delay_duration = 700,
    style = NULL,
    class = NULL
)

```

Arguments

trigger	Single string label rendered on the trigger button.
...	Tooltip content. HTML tags or text are accepted and serialized into the runtime payload.
side	Side relative to the trigger. One of "bottom", "top", "left", "right". Defaults to "top".
align	Alignment along the anchored side. One of "center", "start", "end". Defaults to "center".
delay_duration	Milliseconds to wait after hover/focus before opening. Defaults to 700.
style	Optional inline CSS applied to the tooltip content container (string or named list).
class	Additional classes for the tooltip content container.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

block_value_box	Create a value box
-----------------	--------------------

Description

Create a value box

Usage

```
block_value_box(  
  title,  
  value,  
  ...,  
  description = NULL,  
  icon = NULL,  
  variant = c("default", "accent", "destructive"),  
  class = NULL,  
  style = NULL  
)
```

Arguments

title	Value box title.
value	Primary value.
...	Additional value box body content.
description	Optional value box description.
icon	Optional icon tag or vendored icon name.
variant	Visual variant.
class	Additional classes.
style	Optional inline custom styles.

Value

An `htmltools` tag.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

dismiss_toast	<i>Dismiss a toast notification</i>
---------------	-------------------------------------

Description

Removes a toast from a [block_toaster\(\)](#) before it auto-dismisses. With no `toast_id`, clears every visible toast.

Usage

```
dismiss_toast(  
  session = shiny::getDefaultReactiveDomain(),  
  toaster_id,  
  toast_id = NULL  
)
```

Arguments

<code>session</code>	Shiny session. Defaults to the current reactive session.
<code>toaster_id</code>	Target block_toaster() id.
<code>toast_id</code>	Non-empty id of the toast to dismiss (as returned by show_toast()). NULL dismisses all toasts.

Value

Invisibly returns NULL.

See Also

[block_toaster\(\)](#), [show_toast\(\)](#)

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

dropdown_menu_item *Create a dropdown menu item*

Description

A selectable action row for [block_dropdown_menu\(\)](#). Activating the item reports its value to input\$<id> of the parent menu (event-style, like an action button that carries a value).

Usage

```
dropdown_menu_item(
  value,
  label = value,
  icon = NULL,
  shortcut = NULL,
  disabled = FALSE,
  variant = c("default", "destructive")
)
```

Arguments

value	String reported to the parent menu's input\$<id> when the item is chosen. Required and must be unique within a menu.
label	Item label. Defaults to value.
icon	Optional leading icon: a vendored icon name or an <code>htmltools</code> tag.
shortcut	Optional keyboard-shortcut hint rendered right-aligned (for display only; it does not bind a key).
disabled	Whether the item is non-interactive.
variant	Visual variant. One of "default" or "destructive".

Value

A dropdown-menu item spec consumed by [block_dropdown_menu\(\)](#).

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

dropdown_menu_label	Create a dropdown menu section label
---------------------	--------------------------------------

Description

A non-interactive heading that groups items in a [block_dropdown_menu\(\)](#).

Usage

```
dropdown_menu_label(label)
```

Arguments

label	Section label text.
-------	---------------------

Value

A dropdown-menu label spec consumed by [block_dropdown_menu\(\)](#).

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

dropdown_menu_separator

Create a dropdown menu separator

Description

A horizontal rule that visually divides groups of items in a [block_dropdown_menu\(\)](#).

Usage

```
dropdown_menu_separator()
```

Value

A dropdown-menu separator spec consumed by [block_dropdown_menu\(\)](#).

See Also

Other content: `block_accordion()`, `block_accordion_item()`, `block_alert()`, `block_alert_action()`, `block_alert_description()`, `block_alert_dialog()`, `block_alert_title()`, `block_badge()`, `block_card()`, `block_card_content()`, `block_card_description()`, `block_card_footer()`, `block_card_header()`, `block_card_title()`, `block_code()`, `block_dialog()`, `block_dropdown_menu()`, `block_empty()`, `block_popover()`, `block_progress()`, `block_separator()`, `block_skeleton()`, `block_spinner()`, `block_table()`, `block_toaster()`, `block_tooltip()`, `block_value_box()`, `dismiss_toast()`, `dropdown_menu_item()`, `dropdown_menu_label()`, `inc_block_progress()`, `show_toast()`, `table_column()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`, `update_block_dropdown_menu()`, `update_block_popover()`, `update_block_progress()`, `update_block_table()`, `update_block_toaster()`

inc_block_progress	<i>Increment a runtime progress bar</i>
--------------------	---

Description

Adds amount to a `block_progress()`'s current value, clamped client-side into `[min, max]`. amount may be negative (decrement). Reaching max is a stable no-op state: the bar never auto-hides, auto-resets, or fires a completion event. Reset with `update_block_progress(value = min)`.

Usage

```
inc_block_progress(
  session = shiny::getDefaultReactiveDomain(),
  id,
  amount = 0.1,
  message,
  detail
)
```

Arguments

session	Shiny session. Defaults to the current reactive domain.
id	Component id passed to <code>block_progress()</code> .
amount	Signed amount to add to the current value.
message	Optional status line to set in the same update; NULL clears.
detail	Optional detail text to set in the same update; NULL clears.

Value

Invisibly returns NULL.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

run_showcase

*Run the shinyblocks showcase app***Description**

The showcase is a development-only asset: it ships in the source tree but is excluded from the built package tarball, so this function only works from a source checkout (e.g. via [devtools::load_all\(\)](#)). It errors with a clear message when the app cannot be found.

Usage

```
run_showcase(...)
```

Arguments

... Arguments passed to [shiny::runApp\(\)](#).

Value

The result of [shiny::runApp\(\)](#).

show_toast

*Show a toast notification***Description**

Pushes a transient toast onto a [block_toaster\(\)](#) from the server. Toasts stack, auto-dismiss after duration, and reuse the [block_alert\(\)](#) visual variants and icon system.

Usage

```
show_toast(
  session = shiny::getDefaultReactiveDomain(),
  toaster_id,
  title,
  description = NULL,
  variant = "default",
  icon = "info",
  duration = 5000,
  dismissible = TRUE,
  id = NULL
)
```

Arguments

session	Shiny session. Defaults to the current reactive session.
toaster_id	Target block_toaster() id (unnamed; namespaced via session\$ns()).
title	Toast title. Required.
description	Optional secondary text.
variant	Visual variant. One of "default", "destructive", "success", "warning", "info". Defaults to "default".
icon	Optional icon tag or vendored icon name. Defaults to "info". Pass NULL for no icon.
duration	Finite milliseconds before auto-dismiss. Use 0 (or a negative value) to keep the toast until dismissed. Defaults to 5000.
dismissible	Scalar logical; whether the toast shows a close button. Defaults to TRUE.
id	Optional stable non-empty toast id. Auto-generated when omitted; supply one to target the toast later with dismiss_toast() .

Value

Invisibly returns the toast id.

See Also

[block_toaster\(\)](#), [dismiss_toast\(\)](#)

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

table_column

*Define display options for a block_table() column***Description**

Define display options for a block_table() column

Usage

```
table_column(
  label = NULL,
  align = c("left", "center", "right"),
  format = NULL,
  width = NULL,
  digits = NULL,
  na = NULL,
  intent = NULL,
  emphasis = "text",
  class = NULL,
  style = NULL,
  header_intent = NULL,
  header_emphasis = "text",
  header_class = NULL,
  header_style = NULL,
  cell_intent = NULL,
  cell_emphasis = NULL,
  cell_class = NULL,
  cell_style = NULL
)
```

Arguments

label	Header label. Defaults to the data column name.
align	Text alignment. One of "left", "center", or "right".
format	Optional function applied to the rendered column vector (after any max_rows clipping). The result must have the same length as the input and is coerced to character. When supplied, digits is ignored for this column.
width	Optional CSS width for the column.
digits	Optional non-negative integer for default numeric formatting, overriding the table-level digits for this column.
na	Optional string for missing values, overriding the table-level na for this column.
intent	Optional semantic styling intent applied to every <td> in the column. One of "muted", "primary", "secondary", "destructive", "success", "warning", or "accent". Rendered as theme tokens, never a literal color, so it tracks the active theme preset, light/dark, and style profile automatically.

emphasis	How an intent is rendered. One of "text" (colored foreground, the default), "soft" (tinted background + colored text), or "solid" (filled chip). Has no effect without an intent.
class, style	Escape hatch: additional class / inline style applied to every <td> in the column. You own theme-correctness here — prefer <code>var(--token)</code> over literal colors.
header_intent, header_emphasis, header_class, header_style	Same styling controls applied to the column's <th> header cell.
cell_intent, cell_emphasis, cell_class, cell_style	Vectorized per-cell styling. Each is a function(value) called once with the rendered (unformatted) column vector after any max_rows clipping and must return one entry per rendered row (length-1 results are recycled). cell_intent returns intents (use NA for no styling), cell_emphasis returns emphasis values, cell_class returns classes, and cell_style returns CSS declaration strings (e.g. "color: red"). A single fully named list (e.g. <code>list(color = "var(--primary)")</code>) is treated as one style object applied to every row; for per-row style objects return an unnamed list of named lists. Per-cell results win over the column-level intent / emphasis / class / style.

Value

A table column specification.

See Also

Other content: `block_accordion()`, `block_accordion_item()`, `block_alert()`, `block_alert_action()`, `block_alert_description()`, `block_alert_dialog()`, `block_alert_title()`, `block_badge()`, `block_card()`, `block_card_content()`, `block_card_description()`, `block_card_footer()`, `block_card_header()`, `block_card_title()`, `block_code()`, `block_dialog()`, `block_dropdown_menu()`, `block_empty()`, `block_popover()`, `block_progress()`, `block_separator()`, `block_skeleton()`, `block_spinner()`, `block_table()`, `block_toaster()`, `block_tooltip()`, `block_value_box()`, `dismiss_toast()`, `dropdown_menu_item()`, `dropdown_menu_label()`, `dropdown_menu_separator()`, `inc_block_progress()`, `show_toast()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`, `update_block_dropdown_menu()`, `update_block_popover()`, `update_block_progress()`, `update_block_table()`, `update_block_toaster()`

update_block_accordion

Update an accordion from the server

Description

Opens or closes items of a `block_accordion()` that was given an id, mirroring the click behaviour from the server.

Usage

```
update_block_accordion(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  open,
  notify = TRUE
)
```

Arguments

session	Shiny session. Defaults to the current reactive domain.
input_id	Input id passed to block_accordion() .
open	Item value(s) that should be open after the update. For type = "single" a single value or NULL/character(0) (closes all); for type = "multiple" a character vector.
notify	Whether Shiny should receive an input event after the update.

Value

Invisibly returns NULL.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

update_block_alert_dialog

Update an alert dialog

Description

Update an alert dialog

Usage

```
update_block_alert_dialog(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  open,
  title,
  description,
  confirm_label,
  cancel_label,
  confirm_variant,
  size,
  class,
  style
)
```

Arguments

session	Shiny session.
input_id	Alert dialog input id.
open	Optional boolean open state.
title	Optional replacement title.
description	Optional replacement description.
confirm_label	Optional replacement confirmation label.
cancel_label	Optional replacement cancellation label.
confirm_variant	Optional "default" or "destructive" variant.
size	Optional content size preset.
class	Optional replacement classes, or NULL to clear.
style	Optional replacement inline style, or NULL to clear.

Value

Invisibly returns NULL.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

update_block_button	<i>Update a runtime button</i>
---------------------	--------------------------------

Description

Send a runtime message to a [block_button\(\)](#) created with `id = "..."`. Any argument left unspecified is preserved on the client. Pass `NULL` for `icon` or `style` to clear them.

Usage

```
update_block_button(  
  session = shiny::getDefaultReactiveDomain(),  
  input_id,  
  label,  
  variant,  
  size,  
  icon,  
  icon_position,  
  disabled,  
  style,  
  class  
)
```

Arguments

<code>session</code>	Shiny session. Defaults to the current reactive domain.
<code>input_id</code>	Input id passed to block_button() (via <code>id = "..."</code>).
<code>label</code>	Optional replacement label.
<code>variant</code>	Optional new visual variant.
<code>size</code>	Optional new size.
<code>icon</code>	Optional vendored icon name, <code>shiny.tag</code> , or <code>NULL</code> to clear.
<code>icon_position</code>	Optional <code>"inline-start"</code> / <code>"inline-end"</code> .
<code>disabled</code>	Optional disabled state.
<code>style</code>	Optional inline CSS styles, or <code>NULL</code> to clear.
<code>class</code>	Optional replacement classes for the wrapper.

Value

Invisibly returns `NULL`.

See Also

Other action: [block_button\(\)](#), [block_task_button\(\)](#), [update_block_task_button\(\)](#)

update_block_checkbox *Update a runtime checkbox input*

Description

Update a runtime checkbox input

Usage

```
update_block_checkbox(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  checked,
  disabled,
  style,
  class,
  notify = TRUE
)
```

Arguments

session	Shiny session. Defaults to the current reactive domain.
input_id	Input id passed to block_checkbox().
checked	Optional checked state.
disabled	Optional disabled state.
style	Optional replacement inline CSS styles.
class	Optional replacement classes.
notify	Whether Shiny should receive an input event when checked is updated. Cosmetic-only updates never notify.

Value

Invisibly returns NULL.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

update_block_combobox *Update a runtime combobox input*

Description

Server-side updater for `block_combobox()`. Mirrors `update_block_select()`: `multiple`, `max_items`, and `style` are create-only; when `selected` is supplied without choices, it is validated for shape only and resolved against the client's current choice list.

Usage

```
update_block_combobox(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  choices,
  selected,
  placeholder,
  search_placeholder,
  empty_message,
  disabled,
  width,
  class,
  size,
  invalid,
  notify = TRUE
)
```

Arguments

<code>session</code>	Shiny session. Defaults to the current reactive domain.
<code>input_id</code>	Input id passed to <code>block_select()</code> .
<code>choices</code>	Optional replacement choices.
<code>selected</code>	Optional selected value. Use a character vector for multiple selects. NULL clears single selects to ""; <code>character(0)</code> clears multiple selects.
<code>placeholder</code>	Optional replacement placeholder.
<code>search_placeholder</code>	Optional replacement filter-box placeholder.
<code>empty_message</code>	Optional replacement empty-state message.
<code>disabled</code>	Optional disabled state.
<code>width</code>	Optional replacement CSS width value.
<code>class</code>	Optional replacement classes.
<code>size</code>	Optional replacement size. One of "default", "sm", or "lg".
<code>invalid</code>	Optional invalid/error state.
<code>notify</code>	Whether Shiny should receive an input event when <code>selected</code> is updated. Cosmetic-only updates never notify.

Value

Invisibly returns NULL.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

update_block_date_picker

Update a runtime date picker

Description

Updates the value, bounds, and cosmetic props of a [block_date_picker\(\)](#). Following [shiny::updateDateInput\(\)](#), omitted arguments are left unchanged. To clear the selected date from the server, pass `clear = TRUE` (a bare value = NULL is ignored, matching Shiny's "missing args are ignored" rule).

Usage

```
update_block_date_picker(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  value,
  min,
  max,
  placeholder,
  disabled,
  invalid,
  class,
  style,
  notify = TRUE,
  clear = FALSE
)
```

Arguments

<code>session</code>	Shiny session. Defaults to the current reactive domain.
<code>input_id</code>	Input id passed to block_date_picker() .

value	Optional replacement date (Date, POSIX time, or "yyyy-mm-dd" string).
min	Optional replacement lower bound. Use NULL to clear the bound.
max	Optional replacement upper bound. Use NULL to clear the bound.
placeholder	Optional replacement placeholder text.
disabled	Optional disabled state.
invalid	Optional invalid flag.
class	Optional replacement classes for the wrapper.
style	Optional replacement inline CSS styles for the trigger.
notify	Whether Shiny should receive an input event when value changes. Cosmetic-only updates never notify.
clear	Whether to clear the selected date (sends an explicit empty value). Takes precedence over value.

Value

Invisibly returns NULL.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

update_block_date_range_picker

Update a runtime date range picker

Description

Updates the range, bounds, and cosmetic props of a [block_date_range_picker\(\)](#). Following [shiny::updateDateRangeInput\(\)](#), omitted arguments are left unchanged, and start/end can be updated independently. Note that the control only reports a *complete* range: setting a single endpoint when the other is empty leaves the value NULL (the trigger keeps its placeholder) until both endpoints are present, so the input event fires but `input$<id>` stays empty. To clear the selected range from the server, pass `clear = TRUE` (a bare `start = NULL/end = NULL` is ignored, matching Shiny's "missing args are ignored" rule).

Usage

```
update_block_date_range_picker(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  start,
  end,
  min,
  max,
  separator,
  placeholder,
  disabled,
  invalid,
  class,
  style,
  notify = TRUE,
  clear = FALSE
)
```

Arguments

session	Shiny session. Defaults to the current reactive domain.
input_id	Input id passed to <code>block_date_range_picker()</code> .
start	Optional replacement range start (Date, POSIX time, or "yyyy-mm-dd" string).
end	Optional replacement range end.
min	Optional replacement lower bound. Use NULL to clear the bound.
max	Optional replacement upper bound. Use NULL to clear the bound.
separator	Optional replacement separator text.
placeholder	Optional replacement placeholder text.
disabled	Optional disabled state.
invalid	Optional invalid flag.
class	Optional replacement classes for the wrapper.
style	Optional replacement inline CSS styles for the trigger.
notify	Whether Shiny should receive an input event when the range changes. Cosmetic-only updates never notify.
clear	Whether to clear the selected range (sends an explicit empty range). Takes precedence over start/end.

Value

Invisibly returns NULL.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

update_block_dialog	<i>Update a runtime dialog</i>
---------------------	--------------------------------

Description

Send a server-driven update to a `block_dialog()`.

Usage

```
update_block_dialog(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  open,
  title,
  description,
  footer,
  size,
  class,
  style,
  notify = TRUE
)
```

Arguments

<code>session</code>	Shiny session. Defaults to the current reactive session.
<code>input_id</code>	Dialog input id (unnamed; updaters namespace via <code>session\$ns()</code>).
<code>open</code>	Optional boolean. TRUE opens, FALSE closes.
<code>title</code>	Optional replacement title.
<code>description</code>	Optional replacement description.
<code>footer</code>	Optional replacement footer content. Pass NULL to remove an existing footer.
<code>size</code>	Optional content size: "sm", "default", "lg", "xl".
<code>class</code>	Optional replacement classes for the dialog content container, or NULL to clear.
<code>style</code>	Optional replacement inline CSS styles for the dialog content container, or NULL to clear.
<code>notify</code>	Whether Shiny receives an input event when open changes. Cosmetic-only updates never notify.

Value

Invisibly returns NULL.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

update_block_dropdown_menu

Update a dropdown menu

Description

Send a server-driven update to a [block_dropdown_menu\(\)](#).

Usage

```
update_block_dropdown_menu(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  items,
  open,
  side,
  align,
  disabled,
  style,
  class
)
```

Arguments

session	Shiny session. Defaults to the current reactive session.
input_id	Dropdown-menu input id (unnamed; updaters namespace via <code>session\$ns()</code>).
items	Optional replacement menu parts (list of dropdown_menu_item() / dropdown_menu_label() / dropdown_menu_separator()).
open	Optional boolean. TRUE opens the menu, FALSE closes it.
side	Optional side: "bottom", "top", "left", "right".

align	Optional alignment: "start", "center", "end".
disabled	Optional trigger disabled state.
style	Optional replacement content style (CSS string or named list). Pass NULL to clear.
class	Optional replacement content classes. Pass NULL to clear.

Value

Invisibly returns NULL.

See Also

Other content: `block_accordion()`, `block_accordion_item()`, `block_alert()`, `block_alert_action()`, `block_alert_description()`, `block_alert_dialog()`, `block_alert_title()`, `block_badge()`, `block_card()`, `block_card_content()`, `block_card_description()`, `block_card_footer()`, `block_card_header()`, `block_card_title()`, `block_code()`, `block_dialog()`, `block_dropdown_menu()`, `block_empty()`, `block_popover()`, `block_progress()`, `block_separator()`, `block_skeleton()`, `block_spinner()`, `block_table()`, `block_toaster()`, `block_tooltip()`, `block_value_box()`, `dismiss_toast()`, `dropdown_menu_item()`, `dropdown_menu_label()`, `dropdown_menu_separator()`, `inc_block_progress()`, `show_toast()`, `table_column()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`, `update_block_popover()`, `update_block_progress()`, `update_block_table()`, `update_block_toaster()`

update_block_file_input

Update a runtime file input

Description

Updates the cosmetic and stateful props of a `block_file_input()`. The file value itself is owned by Shiny's native file binding and, as with `shiny::fileInput()`, cannot be set from the server; use `reset = TRUE` to clear the current selection client-side.

Usage

```
update_block_file_input(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  variant,
  button_label,
  placeholder,
  dropzone_label,
  dropzone_hint,
  dropzone_icon,
  dropzone_content,
  accept,
```

```

multiple,
disabled,
invalid,
style,
class,
reset = FALSE
)

```

Arguments

session	Shiny session. Defaults to the current reactive domain.
input_id	Input id passed to <code>block_file_input()</code> .
variant	Optional replacement variant. One of "button" or "dropzone".
button_label	Optional replacement button text.
placeholder	Optional replacement placeholder text.
dropzone_label	Optional replacement dropzone label. Use NULL to clear.
dropzone_hint	Optional replacement dropzone hint. Use NULL to clear.
dropzone_icon	Optional replacement dropzone icon (name string or <code>htmltools</code> tag). Use NULL to clear.
dropzone_content	Optional replacement dropzone interior (<code>htmltools</code> tag/tagList). Use NULL to clear and restore the default icon/label/hint stack.
accept	Optional replacement accepted types. Use NULL to clear.
multiple	Optional flag for allowing multiple files.
disabled	Optional disabled state.
invalid	Optional invalid flag.
style	Optional replacement inline CSS styles for the visible control.
class	Optional replacement classes for the visible control.
reset	Whether to clear the current file selection.

Value

Invisibly returns NULL.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

update_block_input	<i>Update a runtime text input</i>
--------------------	------------------------------------

Description

Update a runtime text input

Usage

```
update_block_input(  
  session = shiny::getDefaultReactiveDomain(),  
  input_id,  
  value,  
  placeholder,  
  type,  
  min,  
  max,  
  step,  
  disabled,  
  invalid,  
  style,  
  class,  
  notify = TRUE  
)
```

Arguments

session	Shiny session. Defaults to the current reactive domain.
input_id	Input id passed to <code>block_input()</code> .
value	Optional replacement value.
placeholder	Optional replacement placeholder text.
type	Optional input type.
min	Optional replacement lower bound (number type only). Use NULL to clear.
max	Optional replacement upper bound (number type only). Use NULL to clear.
step	Optional replacement step size (number type only). Use NULL to reset to the browser default of 1.
disabled	Optional disabled state.
invalid	Optional invalid flag.
style	Optional replacement inline CSS styles for the input.
class	Optional replacement classes for the wrapper.
notify	Whether Shiny should receive an input event when value changes. Cosmetic-only updates never notify.

Value

Invisibly returns NULL.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

update_block_nav	Select a sidebar navigation item from the server
------------------	--

Description

Activates an item rendered by [block_nav_item\(\)](#) inside a [block_nav\(\)](#) that was given an id, mirroring the click behaviour from the server.

Usage

```
update_block_nav(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  selected,
  notify = TRUE
)
```

Arguments

session	Shiny session. Defaults to the current reactive domain.
input_id	Input id passed to block_nav() .
selected	Nav item value to activate.
notify	Whether Shiny should receive an input event after the item is selected.

Value

Invisibly returns NULL.

See Also

Other navigation: [block_breadcrumb\(\)](#), [block_breadcrumb_ellipsis\(\)](#), [block_breadcrumb_item\(\)](#), [block_nav\(\)](#), [block_nav_group\(\)](#), [block_nav_item\(\)](#), [block_nav_label\(\)](#), [block_tab\(\)](#), [block_tabs\(\)](#), [update_block_tabs\(\)](#)

`update_block_pagination`*Update a pagination input*

Description

Reducing pages below the current selection clamps to the new last page. `sibling_count` and `show_edges` are create-only.

Usage

```
update_block_pagination(  
  session = shiny::getDefaultReactiveDomain(),  
  input_id,  
  pages,  
  selected,  
  disabled,  
  style,  
  class,  
  notify = TRUE  
)
```

Arguments

<code>session</code>	Shiny session.
<code>input_id</code>	Input id passed to block_pagination() .
<code>pages</code>	Optional new total number of pages.
<code>selected</code>	Optional new selected page.
<code>disabled</code>	Optional disabled state.
<code>style</code>	Optional replacement inline CSS styles.
<code>class</code>	Optional replacement classes.
<code>notify</code>	Whether Shiny should receive an event when selection changes.

Value

Invisibly returns `NULL`.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#).

update_block_file_input(), update_block_input(), update_block_radio_group(), update_block_select(),
update_block_slider(), update_block_switch(), update_block_textarea(), update_block_toggle_group()

update_block_popover *Update a runtime popover*

Description

Send a server-driven update to a block_popover().

Usage

```
update_block_popover(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  open,
  trigger,
  body,
  side,
  align,
  style,
  class,
  notify = TRUE
)
```

Arguments

session	Shiny session. Defaults to the current reactive session.
input_id	Popover input id (unnamespaced; updaters namespace via session\$ns()).
open	Optional boolean. TRUE opens, FALSE closes.
trigger	Optional replacement trigger label.
body	Optional replacement body content. Pass NULL to clear.
side	Optional side: "bottom", "top", "left", "right".
align	Optional alignment: "center", "start", "end".
style	Optional replacement content style (CSS string or named list). Pass NULL to clear.
class	Optional replacement content classes. Pass NULL to clear.
notify	Whether Shiny receives an input event when open changes. Cosmetic-only updates never notify.

Value

Invisibly returns NULL.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

update_block_progress *Update a runtime progress bar*

Description

Sets fields on a [block_progress\(\)](#) from the server. Following Shiny's `setProgress()`, omitted arguments are left unchanged on the client. Text fields (`message`, `detail`, `label`) clear when passed an explicit `NULL`; numeric fields error on `NULL` (omit them to preserve the current value).

Usage

```
update_block_progress(
  session = shiny::getDefaultReactiveDomain(),
  id,
  value,
  min,
  max,
  message,
  detail,
  label,
  show_value,
  indeterminate,
  variant,
  class,
  style
)
```

Arguments

<code>session</code>	Shiny session. Defaults to the current reactive domain.
<code>id</code>	Component id passed to block_progress() .
<code>value</code>	Replacement progress value (clamped client-side).
<code>min</code>	Replacement lower bound.

max	Replacement upper bound.
message	Replacement status line; NULL clears it.
detail	Replacement detail text; NULL clears it.
label	Replacement label; NULL clears it.
show_value	Whether to render the percent.
indeterminate	Whether to show the unknown-progress sweep.
variant	Replacement indicator color.
class	Replacement classes; NULL clears them.
style	Replacement inline styles; NULL clears them.

Details

Because a partial update cannot see the client’s current state, `min < max` is validated here only when both are supplied in the same call, and `value` is not clamped server-side. The runtime is the single source of truth: it clamps `value` into the merged `[min, max]` and reconciles range validity (see [block_progress\(\)](#)). Supplying `min/max` that invert the client’s current bounds is therefore the caller’s responsibility.

Value

Invisibly returns NULL.

See Also

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_table\(\)](#), [update_block_toaster\(\)](#)

update_block_radio_group
<i>Update a runtime radio group input</i>

Description

Update a runtime radio group input

Usage

```
update_block_radio_group(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  selected,
  choices,
  disabled,
  invalid,
  orientation,
  style,
  class,
  notify = TRUE
)
```

Arguments

session	Shiny session. Defaults to the current reactive domain.
input_id	Input id passed to <code>block_radio_group()</code> .
selected	Optional new selected value.
choices	Optional replacement choices.
disabled	Optional disabled state.
invalid	Optional invalid flag.
orientation	Optional new orientation.
style	Optional replacement inline CSS styles.
class	Optional replacement classes.
notify	Whether Shiny should receive an input event when selected changes. Cosmetic-only updates never notify.

Value

Invisibly returns `NULL`.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

update_block_select	<i>Update a runtime select input</i>
---------------------	--------------------------------------

Description

Not every `block_select()` argument can be changed after creation: `multiple`, `max_items`, and `style` are create-only. Also note that when `selected` is supplied without `choices`, it is validated for shape only — the server cannot see the client's current choice list, so membership is resolved client-side.

Usage

```
update_block_select(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  choices,
  selected,
  placeholder,
  disabled,
  width,
  class,
  size,
  invalid,
  notify = TRUE
)
```

Arguments

<code>session</code>	Shiny session. Defaults to the current reactive domain.
<code>input_id</code>	Input id passed to <code>block_select()</code> .
<code>choices</code>	Optional replacement choices.
<code>selected</code>	Optional selected value. Use a character vector for multiple selects. NULL clears single selects to ""; <code>character(0)</code> clears multiple selects.
<code>placeholder</code>	Optional replacement placeholder.
<code>disabled</code>	Optional disabled state.
<code>width</code>	Optional replacement CSS width value.
<code>class</code>	Optional replacement classes.
<code>size</code>	Optional replacement size. One of "default", "sm", or "lg".
<code>invalid</code>	Optional invalid/error state.
<code>notify</code>	Whether Shiny should receive an input event when <code>selected</code> is updated. Cosmetic-only updates never notify.

Value

Invisibly returns NULL.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_slider()`, `update_block_switch()`, `update_block_textarea()`, `update_block_toggle_group()`

update_block_slider	<i>Update a runtime slider input</i>
---------------------	--------------------------------------

Description

Update a runtime slider input

Usage

```
update_block_slider(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  value,
  min,
  max,
  step,
  orientation,
  show_value,
  min_label,
  max_label,
  disabled,
  invalid,
  style,
  class,
  notify = TRUE
)
```

Arguments

session	Shiny session. Defaults to the current reactive domain.
input_id	Input id passed to <code>block_slider()</code> .
value	Optional replacement value. One or two numeric values.
min	Optional lower bound.
max	Optional upper bound.
step	Optional step size.

orientation	Optional slider orientation. One of "horizontal" or "vertical".
show_value	Optional flag for rendering the current value label.
min_label	Optional replacement minimum label. Use NULL to clear.
max_label	Optional replacement maximum label. Use NULL to clear.
disabled	Optional disabled state.
invalid	Optional invalid flag.
style	Optional replacement inline CSS styles for the slider.
class	Optional replacement classes for the wrapper.
notify	Whether Shiny should receive an input event when value changes. Cosmetic-only updates never notify.

Value

Invisibly returns NULL.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#), [update_block_toggle_group\(\)](#)

update_block_switch	<i>Update a runtime switch input</i>
---------------------	--------------------------------------

Description

Update a runtime switch input

Usage

```
update_block_switch(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  checked,
  disabled,
  size,
  style,
  class,
  notify = TRUE
)
```

Arguments

session	Shiny session. Defaults to the current reactive domain.
input_id	Input id passed to <code>block_switch()</code> .
checked	Optional checked state.
disabled	Optional disabled state.
size	Optional replacement size. One of "default", "sm", or "lg".
style	Optional replacement inline CSS styles.
class	Optional replacement classes.
notify	Whether Shiny should receive an input event when checked is updated. Cosmetic-only updates never notify.

Value

Invisibly returns NULL.

See Also

Other forms: `block_checkbox()`, `block_combobox()`, `block_date_picker()`, `block_date_range_picker()`, `block_field()`, `block_field_description()`, `block_field_group()`, `block_field_invalid()`, `block_field_label()`, `block_field_legend()`, `block_field_set()`, `block_file_input()`, `block_input()`, `block_input_group()`, `block_input_group_addon()`, `block_pagination()`, `block_radio_group()`, `block_select()`, `block_slider()`, `block_switch()`, `block_textarea()`, `block_toggle_group()`, `update_block_checkbox()`, `update_block_combobox()`, `update_block_date_picker()`, `update_block_date_range_picker()`, `update_block_file_input()`, `update_block_input()`, `update_block_pagination()`, `update_block_radio_group()`, `update_block_select()`, `update_block_slider()`, `update_block_textarea()`, `update_block_toggle_group()`

update_block_table	<i>Update a runtime table from the server</i>
--------------------	---

Description

Re-renders a `block_table()` (created with an id) by pushing a freshly formatted payload to the browser. Every argument runs through the same formatting pipeline as `block_table()`, so the refreshed table is identical to one rendered with the same arguments at UI time.

Usage

```
update_block_table(
  session = shiny::getDefaultReactiveDomain(),
  id,
  data = NULL,
  columns = NULL,
  caption = NULL,
  max_rows = NULL,
  na = "",
```

```

    digits = NULL,
    rownames = FALSE,
    row_format = NULL,
    striped = FALSE,
    hover = TRUE,
    bordered = FALSE,
    loading = NULL,
    selection = NULL,
    selected = NULL
  )

```

Arguments

session	Shiny session. Defaults to the current reactive domain.
id	Input id passed to <code>block_table(id =)</code> .
data	Optional replacement data frame. When supplied, the table re-renders with the formatting arguments below.
columns, caption, max_rows, na, digits, rownames, row_format, striped, hover, bordered	Optional formatting arguments, matching <code>block_table()</code> . Used only when data is supplied.
loading	Optional flag. TRUE shows skeleton rows; FALSE clears the loading state without changing data.
selection	Optional new row-selection mode ("none", "single", or "multiple"). NULL leaves the current mode unchanged.
selected	Optional integer vector of 1-based row indices to select. Pass <code>integer(0)</code> to clear the current selection. NULL leaves it unchanged.

Value

Invisibly returns NULL.

See Also

Other content: `block_accordion()`, `block_accordion_item()`, `block_alert()`, `block_alert_action()`, `block_alert_description()`, `block_alert_dialog()`, `block_alert_title()`, `block_badge()`, `block_card()`, `block_card_content()`, `block_card_description()`, `block_card_footer()`, `block_card_header()`, `block_card_title()`, `block_code()`, `block_dialog()`, `block_dropdown_menu()`, `block_empty()`, `block_popover()`, `block_progress()`, `block_separator()`, `block_skeleton()`, `block_spinner()`, `block_table()`, `block_toaster()`, `block_tooltip()`, `block_value_box()`, `dismiss_toast()`, `dropdown_menu_item()`, `dropdown_menu_label()`, `dropdown_menu_separator()`, `inc_block_progress()`, `show_toast()`, `table_column()`, `update_block_accordion()`, `update_block_alert_dialog()`, `update_block_dialog()`, `update_block_dropdown_menu()`, `update_block_popover()`, `update_block_progress()`, `update_block_toaster()`

update_block_tabs	<i>Update styled tabs</i>
-------------------	---------------------------

Description

Selects a tab rendered by [block_tabs\(\)](#) from the server.

Usage

```
update_block_tabs(  
  session = shiny::getDefaultReactiveDomain(),  
  input_id,  
  selected,  
  notify = TRUE  
)
```

Arguments

session	Shiny session. Defaults to the current reactive domain.
input_id	Input id passed to block_tabs() .
selected	Tab value to activate.
notify	Whether Shiny should receive an input event after the tab is selected.

Value

Invisibly returns NULL.

See Also

Other navigation: [block_breadcrumb\(\)](#), [block_breadcrumb_ellipsis\(\)](#), [block_breadcrumb_item\(\)](#), [block_nav\(\)](#), [block_nav_group\(\)](#), [block_nav_item\(\)](#), [block_nav_label\(\)](#), [block_tab\(\)](#), [block_tabs\(\)](#), [update_block_nav\(\)](#)

update_block_task_button	<i>Update a runtime task button</i>
--------------------------	-------------------------------------

Description

Send a runtime message to a [block_task_button\(\)](#). Any argument left unspecified is preserved on the client. Pass NULL for icon, icon_busy, style, or class to clear them.

Usage

```
update_block_task_button(  
  session = shiny::getDefaultReactiveDomain(),  
  input_id,  
  state,  
  label,  
  label_busy,  
  variant,  
  size,  
  icon,  
  icon_busy,  
  icon_position,  
  disabled,  
  style,  
  class  
)
```

Arguments

session	Shiny session. Defaults to the current reactive domain.
input_id	Input id passed to <code>block_task_button()</code> .
state	Optional "ready" or "busy".
label	Optional replacement ready-state label.
label_busy	Optional replacement busy label.
variant	Optional new visual variant.
size	Optional new size.
icon	Optional vendored icon name, <code>shiny.tag</code> , or NULL to clear.
icon_busy	Optional busy icon name, <code>shiny.tag</code> , or NULL to clear.
icon_position	Optional "inline-start" / "inline-end".
disabled	Optional disabled state.
style	Optional inline CSS styles, or NULL to clear.
class	Optional replacement classes merged onto the runtime button element. Pass NULL to clear.

Details

Setting `state = "busy"` takes manual control: an automatic reset scheduled by a click will not release the button. Send `state = "ready"` to release it.

Value

Invisibly returns NULL.

See Also[block_task_button\(\)](#)Other action: [block_button\(\)](#), [block_task_button\(\)](#), [update_block_button\(\)](#)

`update_block_textarea` *Update a runtime textarea input*

Description

Update a runtime textarea input

Usage

```
update_block_textarea(  
  session = shiny::getDefaultReactiveDomain(),  
  input_id,  
  value,  
  placeholder,  
  rows,  
  disabled,  
  invalid,  
  resize,  
  style,  
  class,  
  notify = TRUE  
)
```

Arguments

<code>session</code>	Shiny session. Defaults to the current reactive domain.
<code>input_id</code>	Input id passed to <code>block_textarea()</code> .
<code>value</code>	Optional replacement value.
<code>placeholder</code>	Optional replacement placeholder text.
<code>rows</code>	Optional number of visible rows.
<code>disabled</code>	Optional disabled state.
<code>invalid</code>	Optional invalid flag.
<code>resize</code>	Optional textarea resize behavior. One of "vertical", "none", "both", or "horizontal".
<code>style</code>	Optional replacement inline CSS styles for the textarea.
<code>class</code>	Optional replacement classes for the wrapper.
<code>notify</code>	Whether Shiny should receive an input event when value changes. Cosmetic-only updates never notify.

Value

Invisibly returns NULL.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_toggle_group\(\)](#)

update_block_theme	<i>Update the active theme</i>
--------------------	--------------------------------

Description

Update the active theme

Usage

```
update_block_theme(
  session = shiny::getDefaultReactiveDomain(),
  mode = c("system", "light", "dark")
)
```

Arguments

session	Shiny session.
mode	Theme mode: system, light, or dark.

Value

Invisibly returns NULL.

See Also

Other theme: [block_dark_mode_toggle\(\)](#), [block_style\(\)](#), [block_style_profiles\(\)](#), [block_theme\(\)](#), [block_theme_presets\(\)](#)

update_block_toaster *Update a toaster region*

Description

Send a server-driven update to a [block_toaster\(\)](#). Currently supports moving the region to a different screen position without re-mounting it.

Usage

```
update_block_toaster(
  session = shiny::getDefaultReactiveDomain(),
  toaster_id,
  position
)
```

Arguments

session	Shiny session. Defaults to the current reactive session.
toaster_id	Target block_toaster() id.
position	New screen anchor. One of "top-left", "top-center", "top-right", "bottom-left", "bottom-center", "bottom-right".

Value

Invisibly returns NULL.

See Also

[block_toaster\(\)](#), [show_toast\(\)](#), [dismiss_toast\(\)](#)

Other content: [block_accordion\(\)](#), [block_accordion_item\(\)](#), [block_alert\(\)](#), [block_alert_action\(\)](#), [block_alert_description\(\)](#), [block_alert_dialog\(\)](#), [block_alert_title\(\)](#), [block_badge\(\)](#), [block_card\(\)](#), [block_card_content\(\)](#), [block_card_description\(\)](#), [block_card_footer\(\)](#), [block_card_header\(\)](#), [block_card_title\(\)](#), [block_code\(\)](#), [block_dialog\(\)](#), [block_dropdown_menu\(\)](#), [block_empty\(\)](#), [block_popover\(\)](#), [block_progress\(\)](#), [block_separator\(\)](#), [block_skeleton\(\)](#), [block_spinner\(\)](#), [block_table\(\)](#), [block_toaster\(\)](#), [block_tooltip\(\)](#), [block_value_box\(\)](#), [dismiss_toast\(\)](#), [dropdown_menu_item\(\)](#), [dropdown_menu_label\(\)](#), [dropdown_menu_separator\(\)](#), [inc_block_progress\(\)](#), [show_toast\(\)](#), [table_column\(\)](#), [update_block_accordion\(\)](#), [update_block_alert_dialog\(\)](#), [update_block_dialog\(\)](#), [update_block_dropdown_menu\(\)](#), [update_block_popover\(\)](#), [update_block_progress\(\)](#), [update_block_table\(\)](#)

update_block_toggle_group

Update a runtime toggle group input

Description

type and icon_only are create-only. When selected is supplied without choices, membership is resolved client-side. Replacing choices resets item icons, so pass icons together with choices to keep them.

Usage

```
update_block_toggle_group(
  session = shiny::getDefaultReactiveDomain(),
  input_id,
  selected,
  choices,
  icons,
  disabled,
  variant,
  size,
  style,
  class,
  notify = TRUE
)
```

Arguments

session	Shiny session. Defaults to the current reactive domain.
input_id	Input id passed to block_toggle_group().
selected	Optional new pressed value(s). NULL (single) or character(0) (multiple) releases everything.
choices	Optional replacement choices.
icons	Optional icons for the replacement choices (same shape as in block_toggle_group()). Requires choices.
disabled	Optional TRUE/FALSE for the whole group, or a character vector of choice values to disable individually (which also re-enables the group).
variant	Optional new visual variant.
size	Optional new size.
style	Optional replacement inline CSS styles.
class	Optional replacement classes.
notify	Whether Shiny should receive an input event when selected changes. Cosmetic-only updates never notify.

Value

Invisibly returns NULL.

See Also

Other forms: [block_checkbox\(\)](#), [block_combobox\(\)](#), [block_date_picker\(\)](#), [block_date_range_picker\(\)](#), [block_field\(\)](#), [block_field_description\(\)](#), [block_field_group\(\)](#), [block_field_invalid\(\)](#), [block_field_label\(\)](#), [block_field_legend\(\)](#), [block_field_set\(\)](#), [block_file_input\(\)](#), [block_input\(\)](#), [block_input_group\(\)](#), [block_input_group_addon\(\)](#), [block_pagination\(\)](#), [block_radio_group\(\)](#), [block_select\(\)](#), [block_slider\(\)](#), [block_switch\(\)](#), [block_textarea\(\)](#), [block_toggle_group\(\)](#), [update_block_checkbox\(\)](#), [update_block_combobox\(\)](#), [update_block_date_picker\(\)](#), [update_block_date_range_picker\(\)](#), [update_block_file_input\(\)](#), [update_block_input\(\)](#), [update_block_pagination\(\)](#), [update_block_radio_group\(\)](#), [update_block_select\(\)](#), [update_block_slider\(\)](#), [update_block_switch\(\)](#), [update_block_textarea\(\)](#)

Index

* action

- block_button, 15
- block_task_button, 70
- update_block_button, 89
- update_block_task_button, 111

* content

- block_accordion, 4
- block_accordion_item, 6
- block_alert, 7
- block_alert_action, 8
- block_alert_description, 9
- block_alert_dialog, 9
- block_alert_title, 11
- block_badge, 12
- block_card, 16
- block_card_content, 17
- block_card_description, 18
- block_card_footer, 19
- block_card_header, 19
- block_card_title, 20
- block_code, 23
- block_dialog, 29
- block_dropdown_menu, 31
- block_empty, 32
- block_popover, 53
- block_progress, 54
- block_separator, 58
- block_skeleton, 60
- block_spinner, 62
- block_table, 67
- block_toaster, 74
- block_tooltip, 76
- block_value_box, 78
- dismiss_toast, 79
- dropdown_menu_item, 80
- dropdown_menu_label, 81
- dropdown_menu_separator, 81
- inc_block_progress, 82
- show_toast, 83

- table_column, 85

- update_block_accordion, 86
- update_block_alert_dialog, 87
- update_block_dialog, 95
- update_block_dropdown_menu, 96
- update_block_popover, 102
- update_block_progress, 103
- update_block_table, 109
- update_block_toaster, 115

* forms

- block_checkbox, 21
- block_combobox, 24
- block_date_picker, 26
- block_date_range_picker, 28
- block_field, 33
- block_field_description, 34
- block_field_group, 35
- block_field_invalid, 35
- block_field_label, 36
- block_field_legend, 37
- block_field_set, 38
- block_file_input, 38
- block_input, 44
- block_input_group, 45
- block_input_group_addon, 46
- block_pagination, 51
- block_radio_group, 56
- block_select, 57
- block_slider, 60
- block_switch, 65
- block_textarea, 71
- block_toggle_group, 75
- update_block_checkbox, 90
- update_block_combobox, 91
- update_block_date_picker, 92
- update_block_date_range_picker, 93
- update_block_file_input, 97
- update_block_input, 99
- update_block_pagination, 101

- update_block_radio_group, [104](#)
 - update_block_select, [106](#)
 - update_block_slider, [107](#)
 - update_block_switch, [108](#)
 - update_block_textarea, [113](#)
 - update_block_toggle_group, [116](#)
- * **icon**
 - block_icon, [42](#)
- * **layout**
 - block_body, [13](#)
 - block_cluster, [22](#)
 - block_grid, [40](#)
 - block_header, [41](#)
 - block_page, [50](#)
 - block_sidebar, [59](#)
 - block_stack, [63](#)
- * **navigation**
 - block_breadcrumb, [13](#)
 - block_breadcrumb_ellipsis, [14](#)
 - block_breadcrumb_item, [15](#)
 - block_nav, [47](#)
 - block_nav_group, [47](#)
 - block_nav_item, [48](#)
 - block_nav_label, [49](#)
 - block_tab, [66](#)
 - block_tabs, [69](#)
 - update_block_nav, [100](#)
 - update_block_tabs, [111](#)
- * **outputs**
 - block_image_output, [42](#)
 - block_plot_output, [52](#)
- * **theme**
 - block_dark_mode_toggle, [25](#)
 - block_style, [64](#)
 - block_style_profiles, [65](#)
 - block_theme, [72](#)
 - block_theme_presets, [73](#)
 - update_block_theme, [114](#)
- block_accordion, [4](#)
- block_accordion(), [6–12](#), [17–21](#), [24](#), [30](#), [32](#), [33](#), [54](#), [55](#), [59](#), [60](#), [63](#), [69](#), [75](#), [77–84](#), [86–88](#), [96](#), [97](#), [103](#), [104](#), [110](#), [115](#)
- block_accordion_item, [6](#)
- block_accordion_item(), [4](#), [5](#), [7–12](#), [17–21](#), [24](#), [30](#), [32](#), [33](#), [54](#), [55](#), [59](#), [60](#), [63](#), [69](#), [75](#), [77–84](#), [86–88](#), [96](#), [97](#), [103](#), [104](#), [110](#), [115](#)
- block_alert, [7](#)
- block_alert(), [5](#), [6](#), [8–12](#), [17–21](#), [24](#), [30](#), [32](#), [33](#), [54](#), [55](#), [59](#), [60](#), [63](#), [69](#), [74](#), [75](#), [77–84](#), [86–88](#), [96](#), [97](#), [103](#), [104](#), [110](#), [115](#)
- block_alert_action, [8](#)
- block_alert_action(), [5–7](#), [9–12](#), [17–21](#), [24](#), [30](#), [32](#), [33](#), [54](#), [55](#), [59](#), [60](#), [63](#), [69](#), [75](#), [77–84](#), [86–88](#), [96](#), [97](#), [103](#), [104](#), [110](#), [115](#)
- block_alert_description, [9](#)
- block_alert_description(), [5–8](#), [10–12](#), [17–21](#), [24](#), [30](#), [32](#), [33](#), [54](#), [55](#), [59](#), [60](#), [63](#), [69](#), [75](#), [77–84](#), [86–88](#), [96](#), [97](#), [103](#), [104](#), [110](#), [115](#)
- block_alert_dialog, [9](#)
- block_alert_dialog(), [5–9](#), [11](#), [12](#), [17–21](#), [24](#), [30](#), [32](#), [33](#), [54](#), [55](#), [59](#), [60](#), [63](#), [69](#), [75](#), [77–84](#), [86–88](#), [96](#), [97](#), [103](#), [104](#), [110](#), [115](#)
- block_alert_title, [11](#)
- block_alert_title(), [5–10](#), [12](#), [17–21](#), [24](#), [30](#), [32](#), [33](#), [54](#), [55](#), [59](#), [60](#), [63](#), [69](#), [75](#), [77–84](#), [86–88](#), [96](#), [97](#), [103](#), [104](#), [110](#), [115](#)
- block_badge, [12](#)
- block_badge(), [5–11](#), [17–21](#), [24](#), [30](#), [32](#), [33](#), [54](#), [55](#), [59](#), [60](#), [63](#), [69](#), [75](#), [77–84](#), [86–88](#), [96](#), [97](#), [103](#), [104](#), [110](#), [115](#)
- block_body, [13](#)
- block_body(), [23](#), [41](#), [50](#), [59](#), [63](#)
- block_breadcrumb, [13](#)
- block_breadcrumb(), [14](#), [15](#), [47–49](#), [67](#), [70](#), [100](#), [111](#)
- block_breadcrumb_ellipsis, [14](#)
- block_breadcrumb_ellipsis(), [13–15](#), [47–49](#), [67](#), [70](#), [100](#), [111](#)
- block_breadcrumb_item, [15](#)
- block_breadcrumb_item(), [13](#), [14](#), [47–49](#), [67](#), [70](#), [100](#), [111](#)
- block_button, [15](#)
- block_button(), [32](#), [71](#), [89](#), [113](#)
- block_card, [16](#)
- block_card(), [5–12](#), [18–21](#), [24](#), [30](#), [32](#), [33](#), [54](#), [55](#), [59](#), [60](#), [63](#), [69](#), [75](#), [77–84](#), [86–88](#), [96](#), [97](#), [103](#), [104](#), [110](#), [115](#)
- block_card_content, [17](#)
- block_card_content(), [5–12](#), [17–21](#), [24](#), [30](#), [32](#), [33](#), [54](#), [55](#), [59](#), [60](#), [63](#), [69](#), [75](#),

- [77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_card_description, 18](#)
- [block_card_description\(\), 5–12, 17–21, 24, 30, 32, 33, 54, 55, 59, 60, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_card_footer, 19](#)
- [block_card_footer\(\), 5–12, 17, 18, 20, 21, 24, 30, 32, 33, 54, 55, 59, 60, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_card_header, 19](#)
- [block_card_header\(\), 5–12, 17–19, 21, 24, 30, 32, 33, 54, 55, 59, 60, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_card_title, 20](#)
- [block_card_title\(\), 5–12, 17–20, 24, 30, 32, 33, 54, 55, 59, 60, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_checkbox, 21](#)
- [block_checkbox\(\), 25, 27, 29, 34–38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_cluster, 22](#)
- [block_cluster\(\), 13, 41, 50, 59, 63](#)
- [block_code, 23](#)
- [block_code\(\), 5–12, 17–21, 30, 32, 33, 54, 55, 59, 60, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_combobox, 24](#)
- [block_combobox\(\), 22, 27, 29, 34–38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90–93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_dark_mode_toggle, 25](#)
- [block_dark_mode_toggle\(\), 65, 73, 114](#)
- [block_date_picker, 26](#)
- [block_date_picker\(\), 22, 25, 29, 34–38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_date_range_picker, 28](#)
- [block_date_range_picker\(\), 22, 25, 27, 34–38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_dialog, 29](#)
- [block_dialog\(\), 5–12, 17–21, 24, 32, 33, 54, 55, 59, 60, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_dropdown_menu, 31](#)
- [block_dropdown_menu\(\), 5–12, 17–21, 24, 30, 33, 54, 55, 59, 60, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_empty, 32](#)
- [block_empty\(\), 5–12, 17–21, 24, 30, 32, 54, 55, 59, 60, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_field, 33](#)
- [block_field\(\), 22, 25, 27, 29, 34–38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_field_description, 34](#)
- [block_field_description\(\), 22, 25, 27, 29, 34–38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_field_group, 35](#)
- [block_field_group\(\), 22, 25, 27, 29, 34, 36–38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_field_invalid, 35](#)
- [block_field_invalid\(\), 22, 25, 27, 29, 34, 35, 37, 38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_field_label, 36](#)
- [block_field_label\(\), 22, 25, 27, 29, 34–38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_field_legend, 37](#)
- [block_field_legend\(\), 22, 25, 27, 29, 34–38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_field_set, 38](#)
- [block_field_set\(\), 22, 25, 27, 29, 34–37, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76,](#)

- [90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_file_input, 38](#)
- [block_file_input\(\), 22, 25, 27, 29, 34–38, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 97, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_grid, 40](#)
- [block_grid\(\), 13, 23, 41, 50, 59, 63](#)
- [block_header, 41](#)
- [block_header\(\), 13, 23, 41, 50, 59, 63](#)
- [block_icon, 42](#)
- [block_icon\(\), 6, 76](#)
- [block_image_output, 42](#)
- [block_image_output\(\), 53](#)
- [block_input, 44](#)
- [block_input\(\), 22, 25, 27, 29, 34–38, 40, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_input_group, 45](#)
- [block_input_group\(\), 22, 25, 27, 29, 34–38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_input_group_addon, 46](#)
- [block_input_group_addon\(\), 22, 25, 27, 29, 34–38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_nav, 47](#)
- [block_nav\(\), 14, 15, 48, 49, 67, 70, 100, 111](#)
- [block_nav_group, 47](#)
- [block_nav_group\(\), 14, 15, 47, 49, 67, 70, 100, 111](#)
- [block_nav_item, 48](#)
- [block_nav_item\(\), 14, 15, 47–49, 67, 70, 100, 111](#)
- [block_nav_label, 49](#)
- [block_nav_label\(\), 14, 15, 47–49, 67, 70, 100, 111](#)
- [block_page, 50](#)
- [block_page\(\), 13, 23, 41, 59, 63, 64](#)
- [block_pagination, 51](#)
- [block_pagination\(\), 22, 25, 27, 29, 34–38, 40, 45, 46, 56, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_plot_output, 52](#)
- [block_plot_output\(\), 44](#)
- [block_popover, 53](#)
- [block_popover\(\), 5–12, 17–21, 24, 30, 32, 33, 55, 59, 60, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_progress, 54](#)
- [block_progress\(\), 5–12, 17–21, 24, 30, 32, 33, 54, 59, 60, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_radio_group, 56](#)
- [block_radio_group\(\), 22, 25, 27, 29, 34–38, 40, 45, 46, 51, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_select, 57](#)
- [block_select\(\), 22, 24, 25, 27, 29, 34–38, 40, 45, 46, 51, 56, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105–109, 114, 117](#)
- [block_separator, 58](#)
- [block_separator\(\), 5–12, 17–21, 24, 30, 32, 33, 54, 55, 60, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_sidebar, 59](#)
- [block_sidebar\(\), 13, 23, 41, 50, 63](#)
- [block_skeleton, 60](#)
- [block_skeleton\(\), 5–12, 17–21, 24, 30, 32, 33, 54, 55, 59, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_slider, 60](#)
- [block_slider\(\), 22, 25, 27, 29, 34–38, 40, 45, 46, 51, 56, 58, 66, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_spinner, 62](#)
- [block_spinner\(\), 5–12, 17–21, 24, 30, 32, 33, 54, 55, 59, 60, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_stack, 63](#)
- [block_stack\(\), 13, 23, 41, 50, 59](#)
- [block_style, 64](#)
- [block_style\(\), 26, 50, 65, 73, 114](#)
- [block_style_profiles, 65](#)
- [block_style_profiles\(\), 26, 64, 65, 73, 114](#)
- [block_switch, 65](#)
- [block_switch\(\), 22, 25, 27, 29, 34–38, 40,](#)

- [45, 46, 51, 56, 58, 62, 72, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_tab, 66](#)
- [block_tab\(\), 14, 15, 47–49, 70, 100, 111](#)
- [block_table, 67](#)
- [block_table\(\), 5–12, 17–21, 24, 31–33, 54, 55, 59, 60, 63, 75, 77–84, 86–88, 96, 97, 103, 104, 109, 110, 115](#)
- [block_tabs, 69](#)
- [block_tabs\(\), 14, 15, 47–49, 67, 100, 111](#)
- [block_task_button, 70](#)
- [block_task_button\(\), 16, 89, 111, 113](#)
- [block_textarea, 71](#)
- [block_textarea\(\), 22, 25, 27, 29, 34–38, 40, 45, 46, 51, 56, 58, 62, 66, 76, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 117](#)
- [block_theme, 72](#)
- [block_theme\(\), 26, 64, 65, 73, 114](#)
- [block_theme_presets, 73](#)
- [block_theme_presets\(\), 26, 65, 73, 114](#)
- [block_toaster, 74](#)
- [block_toaster\(\), 5–12, 17–21, 24, 31–33, 54, 55, 59, 60, 63, 69, 77–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_toggle_group, 75](#)
- [block_toggle_group\(\), 22, 25, 27, 29, 34–38, 40, 45, 46, 51, 56, 58, 62, 66, 72, 90, 92, 93, 95, 98, 100, 101, 105, 107–109, 114, 116, 117](#)
- [block_tooltip, 76](#)
- [block_tooltip\(\), 5–12, 17–21, 24, 31–33, 54, 55, 59, 60, 63, 69, 75, 78–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [block_value_box, 78](#)
- [block_value_box\(\), 5–12, 17–21, 24, 31–33, 54, 55, 59, 60, 63, 69, 75, 77, 79–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [devtools::load_all\(\), 83](#)
- [dismiss_toast, 79](#)
- [dismiss_toast\(\), 5–12, 17–21, 24, 31–33, 54, 55, 59, 60, 63, 69, 74, 75, 77, 78, 80–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [dropdown_menu_item, 80](#)
- [dropdown_menu_item\(\), 5–12, 17–21, 24, 31–33, 54, 55, 59, 60, 63, 69, 75, 77–79, 81–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [dropdown_menu_label, 81](#)
- [dropdown_menu_label\(\), 5, 6, 8–12, 17–21, 24, 31–33, 54, 55, 59, 60, 63, 69, 75, 77–80, 82–84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [dropdown_menu_separator, 81](#)
- [dropdown_menu_separator\(\), 5, 6, 8–12, 17–21, 24, 31–33, 54, 55, 59, 60, 63, 69, 75, 77–81, 83, 84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [htmltools::img\(\), 43](#)
- [inc_block_progress, 82](#)
- [inc_block_progress\(\), 5, 6, 8–12, 17–21, 24, 31–33, 54, 55, 59, 60, 63, 69, 75, 77–82, 84, 86–88, 96, 97, 103, 104, 110, 115](#)
- [run_showcase, 83](#)
- [shiny::actionButton\(\), 70](#)
- [shiny::conditionalPanel\(\), 47](#)
- [shiny::dateInput\(\), 26, 29](#)
- [shiny::dateRangeInput\(\), 28](#)
- [shiny::fileInput\(\), 97](#)
- [shiny::imageOutput\(\), 42, 43](#)
- [shiny::numericInput\(\), 45](#)
- [shiny::plotOutput\(\), 52](#)
- [shiny::Progress, 54](#)
- [shiny::renderImage\(\), 42](#)
- [shiny::renderPlot\(\), 52, 53](#)
- [shiny::renderUI\(\), 47](#)
- [shiny::updateDateInput\(\), 92](#)
- [shiny::updateDateRangeInput\(\), 93](#)
- [show_toast, 83](#)
- [show_toast\(\), 5, 6, 8–12, 17–21, 24, 31–33, 54, 55, 59, 60, 63, 69, 74, 75, 77–83, 86–88, 96, 97, 103, 104, 110, 115](#)
- [table_column, 85](#)
- [table_column\(\), 5, 6, 8, 9, 11, 12, 17–21, 24, 31–33, 54, 55, 59, 60, 63, 69, 75, 77–84, 87, 88, 96, 97, 103, 104, 110, 115](#)
- [update_block_accordion, 86](#)

- update_block_accordion(), [5](#), [6](#), [8](#), [9](#), [11](#),
[12](#), [17–21](#), [24](#), [31–33](#), [54](#), [55](#), [59](#), [60](#),
[63](#), [69](#), [75](#), [77–84](#), [86](#), [88](#), [96](#), [97](#),
[103](#), [104](#), [110](#), [115](#)
- update_block_alert_dialog, [87](#)
- update_block_alert_dialog(), [5](#), [6](#), [8](#), [9](#),
[11](#), [12](#), [17–21](#), [24](#), [31–33](#), [54](#), [55](#), [59](#),
[60](#), [63](#), [69](#), [75](#), [77–84](#), [86](#), [87](#), [96](#), [97](#),
[103](#), [104](#), [110](#), [115](#)
- update_block_button, [89](#)
- update_block_button(), [16](#), [71](#), [113](#)
- update_block_checkbox, [90](#)
- update_block_checkbox(), [22](#), [25](#), [27](#), [29](#),
[34–38](#), [40](#), [45](#), [46](#), [51](#), [56](#), [58](#), [62](#), [66](#),
[72](#), [76](#), [92](#), [93](#), [95](#), [98](#), [100](#), [101](#), [105](#),
[107–109](#), [114](#), [117](#)
- update_block_combobox, [91](#)
- update_block_combobox(), [22](#), [25](#), [27](#), [29](#),
[34–38](#), [40](#), [45](#), [46](#), [51](#), [57](#), [58](#), [62](#), [66](#),
[72](#), [76](#), [90](#), [93](#), [95](#), [98](#), [100](#), [101](#), [105](#),
[107–109](#), [114](#), [117](#)
- update_block_date_picker, [92](#)
- update_block_date_picker(), [22](#), [25](#), [27](#),
[29](#), [34–38](#), [40](#), [45](#), [46](#), [51](#), [57](#), [58](#), [62](#),
[66](#), [72](#), [76](#), [90](#), [92](#), [95](#), [98](#), [100](#), [101](#),
[105](#), [107–109](#), [114](#), [117](#)
- update_block_date_range_picker, [93](#)
- update_block_date_range_picker(), [22](#),
[25](#), [27](#), [29](#), [34–38](#), [40](#), [45](#), [46](#), [51](#), [57](#),
[58](#), [62](#), [66](#), [72](#), [76](#), [90](#), [92](#), [93](#), [98](#),
[100](#), [101](#), [105](#), [107–109](#), [114](#), [117](#)
- update_block_dialog, [95](#)
- update_block_dialog(), [5](#), [6](#), [8](#), [9](#), [11](#), [12](#),
[17–21](#), [24](#), [31–33](#), [54](#), [55](#), [59](#), [60](#), [63](#),
[69](#), [75](#), [77–84](#), [86–88](#), [97](#), [103](#), [104](#),
[110](#), [115](#)
- update_block_dropdown_menu, [96](#)
- update_block_dropdown_menu(), [5](#), [7–9](#), [11](#),
[12](#), [17–21](#), [24](#), [31–33](#), [54](#), [55](#), [59](#), [60](#),
[63](#), [69](#), [75](#), [77–84](#), [86–88](#), [96](#), [103](#),
[104](#), [110](#), [115](#)
- update_block_file_input, [97](#)
- update_block_file_input(), [22](#), [25](#), [27](#), [29](#),
[34–38](#), [40](#), [45](#), [46](#), [51](#), [57](#), [58](#), [62](#), [66](#),
[72](#), [76](#), [90](#), [92](#), [93](#), [95](#), [100](#), [102](#), [105](#),
[107–109](#), [114](#), [117](#)
- update_block_input, [99](#)
- update_block_input(), [22](#), [25](#), [27](#), [29](#),
[34–38](#), [40](#), [45](#), [46](#), [51](#), [57](#), [58](#), [62](#), [66](#),
[72](#), [76](#), [90](#), [92](#), [93](#), [95](#), [98](#), [102](#), [105](#),
[107–109](#), [114](#), [117](#)
- update_block_nav, [100](#)
- update_block_nav(), [14](#), [15](#), [47–49](#), [67](#), [70](#),
[111](#)
- update_block_pagination, [101](#)
- update_block_pagination(), [22](#), [25](#), [27](#), [29](#),
[34–38](#), [40](#), [45](#), [46](#), [51](#), [57](#), [58](#), [62](#), [66](#),
[72](#), [76](#), [90](#), [92](#), [93](#), [95](#), [98](#), [100](#), [105](#),
[107–109](#), [114](#), [117](#)
- update_block_popover, [102](#)
- update_block_popover(), [5](#), [7–9](#), [11](#), [12](#),
[17–21](#), [24](#), [31–33](#), [54](#), [55](#), [59](#), [60](#), [63](#),
[69](#), [75](#), [77–84](#), [86–88](#), [96](#), [97](#), [104](#),
[110](#), [115](#)
- update_block_progress, [103](#)
- update_block_progress(), [5](#), [7–9](#), [11](#), [12](#),
[17–21](#), [24](#), [31–33](#), [54](#), [55](#), [59](#), [60](#), [63](#),
[69](#), [75](#), [77–84](#), [86–88](#), [96](#), [97](#), [103](#),
[110](#), [115](#)
- update_block_radio_group, [104](#)
- update_block_radio_group(), [22](#), [25](#), [27](#),
[29](#), [34–38](#), [40](#), [45](#), [46](#), [51](#), [57](#), [58](#), [62](#),
[66](#), [72](#), [76](#), [90](#), [92](#), [93](#), [95](#), [98](#), [100](#),
[102](#), [107–109](#), [114](#), [117](#)
- update_block_select, [106](#)
- update_block_select(), [22](#), [25](#), [27](#), [29](#),
[34–38](#), [40](#), [45](#), [46](#), [51](#), [57](#), [58](#), [62](#), [66](#),
[72](#), [76](#), [90–93](#), [95](#), [98](#), [100](#), [102](#), [105](#),
[108](#), [109](#), [114](#), [117](#)
- update_block_slider, [107](#)
- update_block_slider(), [22](#), [25](#), [27](#), [29](#),
[34–38](#), [40](#), [45](#), [46](#), [51](#), [57](#), [58](#), [62](#), [66](#),
[72](#), [76](#), [90](#), [92](#), [93](#), [95](#), [98](#), [100](#), [102](#),
[105](#), [107](#), [109](#), [114](#), [117](#)
- update_block_switch, [108](#)
- update_block_switch(), [22](#), [25](#), [27](#), [29](#),
[34–38](#), [40](#), [45](#), [46](#), [51](#), [57](#), [58](#), [62](#), [66](#),
[72](#), [76](#), [90](#), [92](#), [93](#), [95](#), [98](#), [100](#), [102](#),
[105](#), [107](#), [108](#), [114](#), [117](#)
- update_block_table, [109](#)
- update_block_table(), [5](#), [7–9](#), [11](#), [12](#),
[17–21](#), [24](#), [31–33](#), [54](#), [55](#), [59](#), [60](#), [63](#),
[67–69](#), [75](#), [77–84](#), [86–88](#), [96](#), [97](#),
[103](#), [104](#), [115](#)
- update_block_tabs, [111](#)
- update_block_tabs(), [14](#), [15](#), [47–49](#), [67](#), [70](#),

100
update_block_task_button, *111*
update_block_task_button(), *16, 70, 71, 89*
update_block_textarea, *113*
update_block_textarea(), *22, 25, 27, 29, 34–38, 40, 45, 46, 51, 57, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 102, 105, 107–109, 117*
update_block_theme, *114*
update_block_theme(), *26, 65, 73*
update_block_toaster, *115*
update_block_toaster(), *5, 7–9, 11, 12, 17–21, 24, 31–33, 54, 55, 59, 60, 63, 69, 75, 77–84, 86–88, 96, 97, 103, 104, 110*
update_block_toggle_group, *116*
update_block_toggle_group(), *22, 25, 27, 29, 34–38, 40, 45, 46, 51, 57, 58, 62, 66, 72, 76, 90, 92, 93, 95, 98, 100, 102, 105, 107–109, 114*