Package 'bones'

November 13, 2026

Type Package

Title Content-Shaped Loading Placeholders for 'shiny' Outputs

Version 0.2.0

Description Shows a grey placeholder in the shape of the content while a 'shiny' output loads, in place of a spinner. The user can thus see what is coming, and not only that something happens. The shape comes from the type of the output. The placeholder keeps the space of the content, so the page does not move when the content arrives. When an output calculates again, the old content stays on the screen, dimmed, and a placeholder does not replace it.

License MIT + file LICENSE

Encoding UTF-8

Language en-GB

Depends R (>= 4.1.0)

Imports htmltools (>= 0.5.4), utils

Suggests bslib, DT, echarts4r, gt, highcharter, plotly, reactable, rhandsontable, shiny (>= 1.7.0), shinytest2, testthat (>= 3.0.0), withr

URL https://github.com/tenmeh/bones, https://tenmeh.github.io/bones/

BugReports https://github.com/tenmeh/bones/issues

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Tanmay Chanda [aut, cre, cph]

Maintainer Tanmay Chanda <tanmaychanda96@gmail.com>

Repository CRAN

Date/Publication 2026-11-13 13:40:11 UTC

Contents

bones_animations 2
bones_auto 3
bones_block 4
bones_defaults 5
bones_dependency 7
bones_error 7
bones_shapes 8
bones_skeleton 9
withBones 10

Index **15**

`bones_animations` *List the valid animations*

Description

Returns the names that the `animation` argument of `withBones()`, `bones_skeleton()` and `bones_defaults()` accepts: `"wave"`, `"pulse"`, `"cascade"`, `"sweep"`, `"none"`. See `bones_defaults()` for what each one does.

Usage

```
bones_animations()
```

Value

A character vector.

See Also

`bones_shapes()`, `bones_defaults()`

Examples

```
bones_animations()
```

bones_auto *Wrap every Shiny output in a UI in a skeleton placeholder*

Description

Gives skeletons to an app with no change to each output. `bones_auto()` looks through a UI and puts each Shiny output in `withBones()`. Each one gets the shape of its output, as with `withBones(output)`: a `plotOutput()` gets a chart, a `tableOutput()` gets rows, a DT table gets the shape of DT, and so on.

Usage

```
bones_auto(ui, ..., exclude = NULL)
```

Arguments

`ui`	A UI: a page, a tag, or a tag list.
`...`	Arguments for `withBones()`, used for every output that is wrapped: for example `stale = FALSE` or `animation = "pulse"`. Not `type`, because each output gets the shape of its own kind. Shape arguments that do not apply to an output do not give a warning.
`exclude`	The ids of outputs to leave alone.

Details

Give it the whole page, or only a part of it:

```
ui <- bones_auto(page_sidebar(...))
ui <- page_sidebar(sidebar = ..., bones_auto(navset_card_tab(...)))
```

Value

`ui`, with each output in a placeholder container. A page keeps its class and its theme.

What it leaves alone

- An output that is already in `withBones()`. To give one output its own shape or options, wrap it yourself: `bones_auto()` keeps that wrapper.
- An output whose id is in `exclude`.
- An inline output, such as `textOutput("n", inline = TRUE)`. The wrapper is a block, so it would break the line of text.

A `uiOutput()` can hold anything, so it gets the text shape. Wrap it yourself to give it another shape, such as `type = "cards"`.

`bones_auto()` sees only the UI that you give it. Outputs that a `renderUI()` makes later are not in it. Call `bones_auto()` on the UI inside that `renderUI()` to give them skeletons too.

Examples

```
if (requireNamespace("shiny", quietly = TRUE)) {
  library(shiny)

  ui <- bones_auto(fluidPage(
    plotOutput("chart"),
    tableOutput("results"),
    textOutput("count", inline = TRUE),         # inline: left alone
    withBones(uiOutput("cards"), type = "cards") # wrapped already: kept
  ))
}
```

`bones_block` *A grey block to build your own placeholder*

Description

The building block of a placeholder that you make yourself. Give the blocks to `withBones()` as `skeleton`, laid out with any tags. Each block takes the colours, the corner radius and the animation of bones, so your placeholder moves and looks like the built-in shapes.

Usage

```
bones_block(width = "100%", height = "0.75rem", shape = c("rect", "circle"))
```

Arguments

`width`	A CSS width, or a number of pixels. `"100%"` fills the width.
`height`	A CSS height, or a number of pixels. The default is the height of one line of text in the built-in shapes.
`shape`	`"rect"` for a block with the corner radius of bones, or `"circle"` for a round block, such as an avatar. Give a circle the same width and height.

Details

The tags at the top level of `skeleton` stack with a small gap. Inside a tag of your own, you set the layout: for example `display: flex` for a row, or `display: grid; gap: 0.5rem` for a column.

Value

An htmltools::tag.

See Also

`withBones()` and its section "Your own placeholder".

Examples

```
# A line of text, 60% wide.
bones_block("60%")

# An avatar.
bones_block(48, 48, shape = "circle")

if (requireNamespace("shiny", quietly = TRUE)) {
  library(shiny)

  # A list of three people: an avatar, a name and a line of text each.
  person <- div(
    style = "display: flex; gap: 0.75rem; align-items: center;",
    bones_block(40, 40, shape = "circle"),
    div(
      style = "flex: 1; display: grid; gap: 0.4rem;",
      bones_block("40%"), bones_block("80%", "0.6rem")
    )
  )
  withBones(uiOutput("people"), skeleton = tagList(person, person, person))
}
```

`bones_defaults` *Set the placeholder defaults for the session*

Description

Sets the defaults that `withBones()` uses for the rest of the session. An argument that is `NULL` does not change its default. An argument that you give to `withBones()` is stronger than a default from here. Call it before you build the UI, for example in `global.R`. The options are read when the UI is built.

Usage

```
bones_defaults(
  animation = NULL,
  color = NULL,
  highlight = NULL,
  radius = NULL,
  speed = NULL,
  stale = NULL,
  delay = NULL,
  min_time = NULL,
  remember = NULL,
  error = NULL,
  learn = NULL
)
```

Arguments

`animation`	How the placeholder moves: • `"wave"`: a band of light moves along each bar. • `"pulse"`: the whole placeholder fades out and in. • `"cascade"`: each bar, cell or line lights up a little after the one before, so a ripple runs through the shape. • `"sweep"`: one band of light crosses the whole placeholder, over every mark at once, the lines and dots of a chart too. • `"none"`: no movement. Users who ask their system to reduce motion always get `"none"`.
`color`	The colour of the placeholder, as a CSS colour. By default it comes from the theme of the page: the text colour of the bslib theme, made faint. A colour given here is stronger than the theme, and than the CSS custom property `--bones-color` set by the app. The "Get started" article lists the custom properties for theming in CSS.
`highlight`	The colour of the band of light in the `"wave"` animation, and of the header row of a table. By default it comes from the theme, as `color` does.
`radius`	The CSS corner radius of the placeholder bars. By default it follows the rounding of the theme.
`speed`	The time of one animation cycle, in seconds.
`stale`	The default for the `stale` argument of `withBones()`.
`delay`, `min_time`	The defaults for the `delay` and `min_time` arguments of `withBones()`, in milliseconds.
`remember`	The default for the `remember` argument of `withBones()`.
`error`	The default for the `error` argument of `withBones()`. `NULL` leaves the option unchanged. Use `FALSE` for the error text of Shiny.
`learn`	The default for the `learn` argument of `withBones()`.

Value

With no arguments, a named list of current values. Unset options use the built-in defaults. `NULL` for `color`, `highlight` and `radius` means that the stylesheet uses the page theme. With arguments, the old values, invisibly, in the form that `options()` uses. Give them to `options()` to restore them. Do not give them to `bones_defaults()`: `NULL` there means "do not change", so an option that had no value before would keep the new value.

Examples

```
old <- bones_defaults(animation = "pulse", radius = "0.5rem")
bones_skeleton("text")

options(old)
```

bones_dependency *The bones HTML dependency*

Description

withBones() and bones_skeleton() attach this for you. It is exported for HTML that you make yourself with the bones-* classes, which needs the stylesheet and the script.

Usage

```
bones_dependency()
```

Value

An htmltools::htmlDependency().

Examples

```
bones_dependency()
```

bones_error *A calm panel for a failed output*

Description

Shows a short message when a wrapped output fails. On the first load, the panel fills the space of the output. After a good load, it becomes a banner above the old content. Messages from validate() pass through. Messages from shiny::safeError() show in the panel even when detail is FALSE, including when shiny.sanitize.errors = TRUE.

Usage

```
bones_error(message = NULL, detail = FALSE)
```

Arguments

message	A single string. NULL uses "This content could not be loaded.".
detail	Show the error message below the main message. The default is FALSE. Use TRUE only for development: when shiny.sanitize.errors = FALSE, raw errors can contain file paths, SQL, or private data.

Value

An htmltools::tag with role="alert".

Examples

```
bones_error()
bones_error("The chart could not be loaded.")
```

---

bones_shapes *List the valid skeleton shapes*

Description

Returns all shapes, or the shapes in one group:

Usage

```
bones_shapes(group = NULL)
```

Arguments

group	A group: "basic", "chart" or "table". NULL (the default) returns all shapes.

Details

- Basic: "text", "cards", "value".
- Chart: "plot", "bar", "line", "scatter", "area", "histogram", "pie", "heatmap", "map", "network", "timeline", "wordcloud".
- Table: "table", "dt", "reactable", "gt", "rhandsontable".

The chart group includes "plot", which uses the bar shape. The table group includes "table" and the table package shapes.

Value

A character vector in the same order as the full list of shapes.

See Also

bones_skeleton(), withBones()

Examples

```
bones_shapes()
bones_shapes("chart")
```

bones_skeleton *Make a skeleton placeholder*

Description

Makes the grey placeholder on its own, with no output in it. You do not need this for a Shiny output, because `withBones()` makes the placeholder for you. Use this function where there is no Shiny output: for example, inside a `renderUI()` while you get data, or to look at a shape while you design a page.

Usage

```
bones_skeleton(
  type = "text",
  rows = 6L,
  cols = 4L,
  lines = 3L,
  n = 3L,
  height = NULL
)
```

Arguments

`type`	The shape: `"text"`, `"table"`, `"plot"`, `"cards"`, `"value"`, `"bar"`, `"line"`, `"scatter"`, `"area"`, `"histogram"`, `"pie"`, `"heatmap"`, `"map"`, `"network"`, `"timeline"`, `"wordcloud"`, `"dt"`, `"reactable"`, `"gt"`, `"rhandsontable"`. Use `bones_shapes()` to list shapes by group.
`rows`	The number of body rows, for `type = "table"` and the table package shapes.
`cols`	The number of columns, for `type = "table"` and the table package shapes.
`lines`	The number of lines, for `type = "text"`.
`n`	The number of cards or values, for `type = "cards"` and `"value"`.
`height`	A CSS height for the placeholder, or a number of pixels. `NULL` gives a default for the type, the same one that `withBones()` uses.

Details

The skeleton uses the animation and the colours from `bones_defaults()`. If you give `rows`, `cols`, `lines` or `n` to a shape that does not use it, a warning names the unused argument.

Value

An htmltools::tag, with the stylesheet attached.

Chart shapes

The chart shapes are `"plot"`, `"bar"`, `"line"`, `"scatter"`, `"area"`, `"histogram"`, `"pie"`, `"heatmap"`, `"map"`, `"network"`, `"timeline"`, `"wordcloud"`. `"plot"` is the same as `"bar"`. `withBones()` finds the shape of a map, network, timeline or word cloud widget from its output. `plotOutput()` cannot say which kind it will hold, because `renderPlot()` decides that later, on the server. So you name the kind yourself.

Table shapes

The table shapes are `"table"`, `"dt"`, `"reactable"`, `"gt"`, `"rhandsontable"`. A table from a table package has a shape that copies that package. `withBones()` finds it from the output by itself.

Examples

```
bones_skeleton("text", lines = 4)
bones_skeleton("table", rows = 8, cols = 5)
bones_skeleton("plot", height = "300px")
bones_skeleton("scatter", height = "300px")
bones_skeleton("dt", rows = 10, cols = 5)
```

---

withBones *Wrap a Shiny output in a skeleton placeholder*

Description

Shows a grey placeholder in the shape of the content while the output loads for the first time. When the output calculates again, the old content stays on the screen and is dimmed. The user is possibly reading that content, and grey boxes in its place would remove information and add none.

Usage

```
withBones(
  ui,
  type = NULL,
  rows = 6L,
  cols = 4L,
  lines = 3L,
  n = 3L,
  height = NULL,
  animation = getOption("bones.animation", "wave"),
  stale = getOption("bones.stale", TRUE),
  delay = getOption("bones.delay", 300),
  min_time = getOption("bones.min_time", 500),
  remember = getOption("bones.remember", TRUE),
  skeleton = NULL,
  error = getOption("bones.error", bones_error()),
  learn = getOption("bones.learn", TRUE),
```

```
  retry = NULL,
  hold = NULL,
  speed = getOption("bones.speed"),
  color = getOption("bones.color")
)
```

Arguments

ui	A Shiny output, such as `plotOutput("chart")`.
type	The shape of the skeleton: `"text"`, `"table"`, `"plot"`, `"cards"`, `"value"`, `"bar"`, `"line"`, `"scatter"`, `"area"`, `"histogram"`, `"pie"`, `"heatmap"`, `"map"`, `"network"`, `"timeline"`, `"wordcloud"`, `"dt"`, `"reactable"`, `"gt"`, `"rhandsontable"`. `NULL` (the default) gets the shape from `ui`. Use `bones_shapes()` to list shapes by group.
rows, cols	The number of body rows and columns, when `type` is `"table"` or a table package shape.
lines	The number of lines, when `type` is `"text"`.
n	The number of cards or values, when `type` is `"cards"` or `"value"`.
height	A CSS height for the placeholder, or a number of pixels, as in Shiny. `NULL` uses the height of the output. If the output has no height, `NULL` uses a default for the type. With `skeleton` and an output that has no height, `NULL` uses the natural height of your skeleton in the browser. A height that you give always wins.
animation	`"wave"`, `"pulse"`, `"cascade"`, `"sweep"` or `"none"`. See `bones_defaults()` for what each one does. `NULL` uses the value from `bones_defaults()`.
stale	`TRUE` keeps the old content on the screen, dimmed, while the output calculates again. `FALSE` shows the skeleton again. `NULL` uses the value from `bones_defaults()`, which is `TRUE` if you did not set it.
delay	The time in milliseconds before the skeleton, or the dimming, appears. A load that ends sooner shows nothing, so a fast output does not flicker. The space of the content is kept from the start. `NULL` uses the value from `bones_defaults()`, which is 300 if you did not set it.
min_time	The shortest time in milliseconds that the skeleton, or the dimming, stays on the screen once it has appeared. A load that ends a moment after `delay` thus does not flash a skeleton for one frame. `NULL` uses the value from `bones_defaults()`, which is 500 if you did not set it.
remember	`TRUE` stores the real height of the content in the browser, and keeps that height on the next visit in place of the estimate, so the page does not move when the content arrives. Only a height from an estimate is stored: a height from `height`, or from the output itself, is exact already. The height is kept per page and per output id, and it is used only when the output has almost the same width as when it was measured. For a plotly, echarts4r or highcharter output, `remember` also stores the kind of the chart (see "Shape"). It also allows `learn`. With `skeleton`, its measured height is an estimate too. `remember = FALSE` disables learning, even if `learn = TRUE`. `NULL` uses the value from `bones_defaults()`, which is `TRUE` if you did not set it.

skeleton — Your own placeholder, in place of a built-in shape: a tag or a tag list, made of bones_block() blocks and any layout around them. See "Your own placeholder". Give type or skeleton, not both. It keeps its own look by default. Give learn = TRUE explicitly to use it only until the browser has learned the content for a later visit.

error — A panel for a failed output: a tag or a tag list. NULL uses bones_defaults(), or bones_error() if unset. FALSE keeps the error text of Shiny. Silent req() errors and validate() messages keep their usual behaviour. After a good load, the old content stays dimmed with the panel as a banner.

learn — TRUE stores the shape of the content in the browser of the user, and the first skeleton of the next visit has that shape. Only numbers are stored: the place and the size of each line of text, image and frame. It applies only when you do not give type, rows, cols, lines or n, and not to charts. With skeleton, only an explicit learn = TRUE enables learning; a session default does not. Unused shape arguments do not prevent a custom skeleton from learning. It needs remember = TRUE. A custom skeleton gives a warning if learning is requested with remember = FALSE or for a chart. NULL uses the value from bones_defaults(), which is TRUE if you did not set it.

retry — A single non-empty input id, or NULL for no retry button. Adds "Try again" after the error panel content. The output must read this input to run again. Cannot be used with error = FALSE.

hold — Input ids that show the skeleton at once when they change, or NULL. Use ns() for ids inside modules, for both arguments. A hold hides content but keeps its space, overrides stale and delay, and respects min_time when the load finishes. If this output does not start calculating, the hold releases when Shiny is idle. In JavaScript, Bones.hold(el) holds until the next finished load or until Bones.release(el), even across idle events. el can be the wrapper, the output element, or its id string.

speed — The time of one animation cycle for this output, in seconds. NULL uses bones_defaults(), or 1.4 seconds if unset.

color — The colour of this placeholder, as a CSS colour. NULL uses bones_defaults(), or the theme of the page if unset.

Value

ui, in a placeholder container.

Shape

The shape comes from the output that you give: plotOutput() gets the shape of a chart, tableOutput() gets rows and columns, and textOutput() gets lines of text. Use type to set a different shape. You must do this for uiOutput(), because its content is not known before it arrives. If you give rows, cols, lines or n to a shape that does not use it, a warning names the unused argument. The shape known in R decides this, before any chart kind is found in the browser. None of these arguments apply to a custom skeleton.

A plotOutput() gets the shape of a bar chart. If the chart is of another kind, give that kind: "line", "scatter", "area", "histogram", "pie" or "heatmap". The output cannot tell, because renderPlot() decides the kind later, on the server.

A table from a table package gets the shape of that package, found from its output: `DT::DTOutput()` gets `"dt"`, `reactable::reactableOutput()` gets `"reactable"`, `gt::gt_output()` gets `"gt"`, and `rhandsontable::rHandsontableOutput()` gets `"rhandsontable"`. Each shape copies the parts of its package, such as the search box of DT. Set `rows` to the number of rows on a page: DT and reactable show 10 by default. Before gt 1.0.0, `gt_output()` did not mark its output as a gt table, so with an older gt, give `type = "gt"`.

A visualisation widget gets the shape of its kind, found from its output in the same way. A map (leaflet, tmap, mapview, mapdeck and others) gets `"map"`. A network or a diagram (visNetwork, DiagrammeR, networkD3, collapsibleTree) gets `"network"`. timevis gets `"timeline"`, wordcloud2 gets `"wordcloud"`, and dygraphs gets `"line"`. ggmap and a ggplot2 map draw into a `plotOutput()`, so give them `type = "map"`.

A plotly, echarts4r or highcharter output can draw any kind of chart. With no `type`, it finds its kind by itself, but only when its first value arrives in the browser: the spec of the chart names the kind of each series. The first skeleton thus has the bar shape. The next loads, with `stale = FALSE`, show the shape of the kind: `"bar"`, `"line"`, `"area"`, `"scatter"`, `"histogram"`, `"pie"`, `"heatmap"`, `"map"`, `"network"` or `"wordcloud"`. With `remember`, the kind is also stored in the browser, so on the next visit the first skeleton has that shape too. The first series that names a kind decides. A chart of a kind with no shape, such as a box plot, keeps the bar shape. A `type` that you give always wins. Other widgets that can draw any chart, such as ggiraph, keep the bar shape: give their kind with `type`.

Your own placeholder

When no built-in shape fits, give your own with `skeleton`. Build it from `bones_block()`, a grey block that takes the colours and the animation of bones, and lay the blocks out with any tags. The tags at the top level stack with a small gap. The placeholder behaves as a built-in shape does: it waits for `delay`, it stays for `min_time`, it keeps the space of the content, and the old content stays when `stale` is `TRUE`.

Layout

The wrapper keeps the space of the output until the content arrives, so the page does not move. If the output sets its own height, as `plotOutput()` does, the wrapper uses that height. If not, the height comes from `rows`, `lines` or `n`, or you can set it with `height`. When the content arrives, the content sets the height.

Examples

```
if (requireNamespace("shiny", quietly = TRUE)) {
  library(shiny)

  withBones(plotOutput("chart"))
  withBones(tableOutput("results"), rows = 8, cols = 5)
  withBones(uiOutput("cards"), type = "cards", n = 4)
  withBones(plotOutput("trend"), type = "line")

  # Your own placeholder: an avatar and two lines of text.
  withBones(
    uiOutput("profile"),
```

```
    skeleton = div(
      style = "display: flex; gap: 1rem; align-items: center;",
      bones_block("3rem", "3rem", shape = "circle"),
      div(
        style = "flex: 1; display: grid; gap: 0.5rem;",
        bones_block("60%"), bones_block("40%")
      )
    )
  )
}
```

Index

bones_animations, 2
bones_auto, 3
bones_block, 4
bones_block(), *12, 13*
bones_defaults, 5
bones_defaults(), *2*, *9*, *11, 12*
bones_dependency, 7
bones_error, 7
bones_error(), *12*
bones_shapes, 8
bones_shapes(), *2*, *9*, *11*
bones_skeleton, 9
bones_skeleton(), *2*, *7, 8*

htmltools::htmlDependency(), *7*
htmltools::tag, *4*, *7*, *9*

options(), *6*

withBones, 10
withBones(), *2–10*