

Package: animejs (via r-universe)

June 1, 2026

Title R Bindings to the 'Anime.js' Animation Library

Version 0.1.1

Description Provides low-level R bindings to the 'Anime.js' library (<<https://animejs.com>>), enabling the creation of browser-native SVG and HTML animations via the 'htmlwidgets' framework.

License MIT + file LICENSE

URL <https://github.com/long39ng/animejs>,
<https://long39ng.github.io/animejs/>

BugReports <https://github.com/long39ng/animejs/issues>

Depends R (>= 4.1.0)

Imports htmlwidgets, rlang

Suggests htmltools, jsonlite, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/Needs/website rmarkdown

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/pak/sysreqs cmake make libuv1-dev

Repository <https://long39ng.r-universe.dev>

Date/Publication 2026-04-02 13:51:36 UTC

RemoteUrl <https://github.com/long39ng/animejs>

RemoteRef HEAD

RemoteSha 36aa7e585fd3be70b0d5232cddb403a2e42fe2a3

Contents

anime_add	2
anime_easing	3
anime_from_to	4
anime_keyframes	5
anime_on	6
anime_playback	7
anime_render	8
anime_stagger	9
anime_target_class	10
anime_target_css	10
anime_target_id	11
anime_target_layer	12
anime_timeline	12
animejs_widget	13

Index	14
--------------	-----------

anime_add	<i>Add an animation segment to a timeline</i>
-----------	---

Description

Pipe-friendly. Each call appends one segment: a set of CSS property animations applied to a target selector.

Usage

```
anime_add(
  timeline,
  selector,
  props,
  offset = "+=0",
  duration = NULL,
  ease = NULL,
  delay = NULL,
  stagger = NULL
)
```

Arguments

timeline	An anime_timeline object.
selector	CSS selector string identifying the SVG/HTML elements to animate. Use anime_target_*() helpers to construct selectors.
props	A named list of property animations. Values may be scalars, two-element vectors (from/to), or anime_keyframes() objects.

offset	Timeline offset. "+=N" means N ms after the previous segment ends; a bare number is an absolute position in ms.
duration	Overrides the timeline default for this segment.
ease	Overrides the timeline default for this segment.
delay	Overrides the timeline default for this segment.
stagger	An anime_stagger object for per-element delay offsets.

Value

The modified anime_timeline object.

Examples

```
anime_timeline() |>
  anime_add(
    selector = anime_target_class("circle"),
    props    = list(opacity = anime_from_to(0, 1)),
    duration = 600
  )
```

 anime_easing

Easing constructors

Description

A family of constructors for Anime.js v4 easing specifications. Each returns an anime_easing object that serialises to the correct JS string inside [anime_timeline\(\)](#), [anime_add\(\)](#), or [anime_playback\(\)](#).

Usage

```
anime_easing(family = "Quad", direction = "out")
```

```
anime_easing_elastic(direction = "out", amplitude = 1, period = 0.3)
```

```
anime_easing_back(direction = "out", overshoot = 1.70158)
```

```
anime_easing_bezier(x1, y1, x2, y2)
```

```
anime_easing_steps(count)
```

```
anime_easing_spring(bounce = 0.5, duration = 628)
```

Arguments

family	Character. One of "linear", "Quad", "Cubic", "Quart", "Quint", "Sine", "Expo", "Circ", "Bounce".
direction	Character. One of "in", "out", "inOut", "outIn".
amplitude, period	(Elastic easing) Numeric. Overshoot amplitude and oscillation period.
overshoot	(Back easing) Numeric. Overshoot amount.
x1, y1, x2, y2	(Cubic bezier easing) Coordinates of the first and second control point. x1 and x2 must be in [0, 1].
count	(Steps easing) Positive integer. Number of discrete steps.
bounce	(Spring easing) Number in [-1, 1]. Controls bounciness. Values from 0 to 1 produce bouncy curves; values below 0 produce over-damped curves. Keep within [-0.5, 0.5] for predictable behaviour.
duration	(Spring easing) Number in [10, 10000]. The perceived duration in milliseconds at which the animation feels visually complete.

Value

An anime_easing object.

Examples

```
anime_easing("linear")
anime_easing("Quad", "outIn")

anime_easing_elastic()
anime_easing_elastic("in", amplitude = 1.5, period = 0.3)

anime_easing_back()
anime_easing_back("in", overshoot = 2.5)

anime_easing_bezier(0.4, 0, 0.2, 1)
anime_easing_bezier(0.68, -0.55, 0.265, 1.55)

anime_easing_steps(10)

anime_easing_spring()
anime_easing_spring(bounce = 0.65, duration = 350)
```

anime_from_to

Specify a from/to property range

Description

Convenience constructor for a two-value property animation that runs from from to to. An optional CSS unit suffix is concatenated into both values during serialisation (e.g. 100 with unit = "px" becomes "100px").

Usage

```
anime_from_to(from, to, unit = "")
```

Arguments

from	Numeric. Starting value.
to	Numeric. Ending value.
unit	Character. Optional CSS unit suffix, e.g. "px", "%", "deg".

Value

An anime_from_to object.

Examples

```
anime_from_to(0, 1)
anime_from_to(0, 360, unit = "deg")
```

anime_keyframes	<i>Specify per-property keyframes for an animation</i>
-----------------	--

Description

Constructs a keyframes object for use in the props argument of [anime_add\(\)](#). Each positional argument is one keyframe.

Usage

```
anime_keyframes(...)
```

Arguments

...	Keyframe values. Either bare numeric values, or lists each with a \$to key and optional \$ease and \$duration overrides.
-----	--

Value

An anime_keyframes object.

Examples

```

# Bare numeric keyframe values
anime_add(
  anime_timeline(),
  selector = ".circle",
  props = list(
    opacity = anime_keyframes(0, 1, 0.5),
    translateY = anime_keyframes(-20, 0, 10)
  )
)

# Per-keyframe lists with optional ease and duration overrides
anime_add(
  anime_timeline(),
  selector = ".circle",
  props = list(
    opacity = anime_keyframes(
      list(to = 0),
      list(to = 1, ease = anime_easing("Cubic"), duration = 400),
      list(to = 0.5, ease = "linear", duration = 200)
    )
  )
)

```

`anime_on`*Attach a JavaScript callback to a timeline event*

Description

The callback must be the name of a globally scoped JavaScript function already present on the page, for example one injected via `htmltools::tags$script()`. At render time the JavaScript binding resolves the name to `window[callback]` and attaches it to the corresponding Anime.js timeline hook.

Usage

```

anime_on(
  timeline,
  event = c("onBegin", "onUpdate", "onComplete", "onLoop"),
  callback
)

```

Arguments

<code>timeline</code>	An <code>anime_timeline</code> object.
<code>event</code>	One of "onBegin", "onUpdate", "onComplete", "onLoop", matching the Anime.js v4 timeline callback API.
<code>callback</code>	Character scalar. Name of the global JS function to invoke.

Value

The modified anime_timeline object.

Examples

```
if (interactive() && rlang::is_installed("htmltools")) {
  svg_src <- '<svg viewBox="0 0 200 100" xmlns="http://www.w3.org/2000/svg">
    <circle class="circle" cx="100" cy="50" r="20" fill="#4e79a7"/>
  </svg>'

  widget <- anime_timeline(duration = 800) |>
    anime_add(selector = ".circle", props = list(opacity = c(0, 1))) |>
    anime_on("onComplete", "handleAnimationDone") |>
    anime_render(svg = svg_src)

  callback_js <- htmltools::tags$script(
    "function handleAnimationDone() {
      console.log('Animation complete.');"
    }"
  )

  htmltools::browsable(htmltools::tagList(callback_js, widget))
}
```

anime_playback

Configure timeline playback

Description

Sets autoplay, loop, direction, and optional controls UI on an anime_timeline. Calling this function overwrites any playback settings already on the timeline (including the loop value set in [anime_timeline\(\)](#)).

Usage

```
anime_playback(
  timeline,
  autoplay = TRUE,
  loop = NULL,
  reversed = FALSE,
  alternate = FALSE,
  controls = FALSE
)
```

Arguments

timeline	An anime_timeline object.
autoplay	Logical. Start playing immediately on load.
loop	Logical or integer. FALSE for no looping, TRUE for infinite looping, or a positive integer for a fixed number of iterations.
reversed	Logical. Play the timeline in reverse from the end.
alternate	Logical. Alternate direction on each iteration (requires loop to be TRUE or a positive integer to have any visible effect).
controls	Logical. Inject a play/pause/seek control bar into the widget container.

Value

The modified anime_timeline object.

anime_render	<i>Render an anime_timeline as an htmlwidget</i>
--------------	--

Description

Serialises an [anime_timeline\(\)](#) object to JSON and wraps it together with an SVG payload in an htmlwidget.

Usage

```
anime_render(
  timeline,
  svg = NULL,
  width = NULL,
  height = NULL,
  elementId = NULL
)
```

Arguments

timeline	An anime_timeline object produced by anime_timeline() .
svg	Character. Raw SVG markup to embed in the widget. If NULL, an empty string is used (the timeline will animate against existing DOM content – advanced use only).
width	Fixed width for widget (in css units). The default is NULL, which results in intelligent automatic sizing based on the widget’s container.
height	Fixed height for widget (in css units). The default is NULL, which results in intelligent automatic sizing based on the widget’s container.
elementId	Use an explicit element ID for the widget (rather than an automatically generated one). Useful if you have other JavaScript that needs to explicitly discover and interact with a specific widget instance.

Value

An object of class `c("animejs", "htmlwidget")`.

Examples

```
tl <- anime_timeline(duration = 800) |>
  anime_add(
    selector = anime_target_class("dot"),
    props = list(opacity = anime_from_to(0, 1))
  )
svg <- '<svg viewBox="0 0 100 100"><circle class="dot" cx="50" cy="50" r="10"/></svg>'
if (interactive()) {
  anime_render(tl, svg)
}
```

 anime_stagger

Create a stagger configuration for per-element delay offsets

Description

When applied to a multi-element selector, Anime.js distributes animation start times across elements according to the stagger value.

Usage

```
anime_stagger(value, from = "first", grid = NULL, axis = NULL, ease = NULL)
```

Arguments

value	Numeric. Base delay in milliseconds between each element.
from	One of "first", "last", "center", or a numeric index. Controls which element starts first.
grid	Integer vector of length 2 (<code>c(rows, cols)</code>) for 2D grid stagger.
axis	One of "x", "y". Used together with grid.
ease	Easing applied to the stagger distribution itself.

Value

An `anime_stagger` object.

Examples

```
# Simple linear stagger, 100 ms between elements
anime_stagger(100)

# Stagger from centre outward
anime_stagger(200, from = "center")

# 2-D grid stagger along the x axis
anime_stagger(50, grid = c(3, 4), axis = "x")
```

anime_target_class *Target elements by CSS class*

Description

Target elements by CSS class

Usage

```
anime_target_class(cls)
```

Arguments

cls Character scalar. Class name without a leading dot.

Value

A CSS selector string of the form ".<cls>".

Examples

```
anime_target_class("circle")
```

anime_target_css *Target elements by an arbitrary CSS selector*

Description

A pass-through for selectors not covered by the other anime_target_*() helpers.

Usage

```
anime_target_css(selector)
```

Arguments

selector Character scalar. A valid CSS selector string.

Value

selector unchanged.

Examples

```
anime_target_css(".panel > circle")
```

anime_target_id	<i>Target SVG or HTML elements by a data-animejs-id attribute</i>
-----------------	---

Description

The primary mechanism for targeting individual elements annotated by `svg_annotate()` (in `gganime`) or by hand.

Usage

```
anime_target_id(id)
```

Arguments

id Character scalar. Value of the `data-animejs-id` attribute.

Value

A CSS selector string of the form "[data-animejs-id='<id>']".

Examples

```
anime_target_id("c1")
```

anime_target_layer *Target all data elements belonging to a ggplot2 layer*

Description

Produces an attribute selector matching the data-layer attribute injected by gganime's SVG annotation pipeline. Exposed for power users who compose animejs timelines against annotated ggplot2 SVG output directly.

Usage

```
anime_target_layer(layer_index)
```

Arguments

layer_index Integer scalar. 1-based index of the ggplot2 layer.

Value

A CSS selector string of the form "[data-layer='<layer_index>']".

Examples

```
anime_target_layer(1L)
```

anime_timeline *Initialise an Anime.js timeline*

Description

Initialise an Anime.js timeline

Usage

```
anime_timeline(duration = 1000, ease = anime_easing(), delay = 0, loop = FALSE)
```

Arguments

duration Default duration in milliseconds for all segments.

ease Default easing for all segments.

delay Default delay in milliseconds between segments.

loop Logical or integer. FALSE for no looping, TRUE for infinite looping, or a positive integer for a fixed number of iterations.

Value

An anime_timeline object.

Examples

```
anime_timeline(duration = 800, ease = anime_easing())
```

animejs_widget	<i>Create a bare animejs htmlwidget</i>
----------------	---

Description

This is the low-level constructor. Most users will not call this directly; it is the final rendering step called by [anime_render\(\)](#).

Usage

```
animejs_widget(
  svg,
  timeline_config,
  width = NULL,
  height = NULL,
  elementId = NULL
)
```

Arguments

svg	Character. Raw SVG markup to embed in the widget. If NULL, an empty string is used (the timeline will animate against existing DOM content – advanced use only).
timeline_config	List. A serialisable timeline specification produced by anime_timeline() and its modifiers, then passed through timeline_to_json_config() .
width	Fixed width for widget (in css units). The default is NULL, which results in intelligent automatic sizing based on the widget's container.
height	Fixed height for widget (in css units). The default is NULL, which results in intelligent automatic sizing based on the widget's container.
elementId	Use an explicit element ID for the widget (rather than an automatically generated one). Useful if you have other JavaScript that needs to explicitly discover and interact with a specific widget instance.

Value

An object of class `c("animejs", "htmlwidget")`

Index

anime_add, 2
anime_add(), 3, 5
anime_easing, 3
anime_easing_back (anime_easing), 3
anime_easing_bezier (anime_easing), 3
anime_easing_elastic (anime_easing), 3
anime_easing_spring (anime_easing), 3
anime_easing_steps (anime_easing), 3
anime_from_to, 4
anime_keyframes, 5
anime_on, 6
anime_playback, 7
anime_playback(), 3
anime_render, 8
anime_render(), 13
anime_stagger, 9
anime_target_class, 10
anime_target_css, 10
anime_target_id, 11
anime_target_layer, 12
anime_timeline, 12
anime_timeline(), 3, 7, 8, 13
animejs_widget, 13