Package 'waysign'

May 8, 2026

Title Multi-Purpose and High-Performance Routing

Version 0.1.1

Description Provides routing based on the 'path-tree' 'Rust' crate. The routing is general purpose in the sense that any type of R object can be associated with a path, not just a handler function.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Config/rextendr/version 0.4.2.9000

SystemRequirements Cargo (Rust's package manager), rustc >= 1.65.0, xz

Depends R (>= 4.2)

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Imports rlang (>= 1.1.0)

URL https://github.com/thomasp85/waysign

BugReports https://github.com/thomasp85/waysign/issues

NeedsCompilation yes

Author Thomas Lin Pedersen [aut, cre] (ORCID: <https://orcid.org/0000-0002-5147-4711>), Posit, PBC [cph, fnd] (ROR: <https://ror.org/03wc8by49>)

Maintainer Thomas Lin Pedersen <thomas.pedersen@posit.co>

Repository CRAN

Date/Publication 2026-01-13 09:40:02 UTC

Contents

path_params *Deconstruct a path pattern*

Description

This function parses a path pattern and returns both the name of the parameters and a version of the path formatted for glue string interpolation.

Usage

```
path_params(path)
```

Arguments

path	The path pattern to parse

Value

A list with the elements `keys` containing the names of all the path parameters and `glue` containing a glue ready version of the path

Examples

```
path_params("/users/:user/assets/*")
```

signpost *A simple, multi-purpose router*

Description

The `signpost` class implements a high-performance, multipurpose router build on top of the path-tree library. A router associates filepath-like patterns with a piece of data for latter retrieval. Often that data is a function and the path to be matched against the pattern comes from a URL, but it can be anything, adapting to the need of the user. Objects of the class uses reference semantics so they do not get copied and alterations will affect all instances of the object.

Usage

```
signpost()
```

Details

The path pattern supported by Waysign mirrors that of path-tree and while the full documentation can be found there, it will be briefly explained here.

A path pattern consist of zero, one, or more elements separated by `/` (always started by `/`). Each element can either be a literal or one of the following variable types:

- `:name` matches a single path piece
- `:name?` matches an optional path piece
- `:name+` or `+` matches one or more path pieces
- `:name*` or `*` matches zero or more path pieces

A variable don't have to consume a full path element. E.g. you could have a path pattern like this: `/date/:day-:month-:year` which would match to paths such as `/date/24-12-2025`

Value

A `waysign` router. See the *Methods* section for a description of its behavior

Methods

`add_path(path, object)`:

Add a new path to the router. See *Details* for allowed path syntax

Arguments:

- `path`: A string giving the path to add
- `object`: An R object to be associated with the path

Returns:
The object, invisibly

`find_object(path)`:

Search for a path in the router

Arguments:

- `path`: The path to search for

Returns:
If no matching path is found then `NULL`, otherwise a list with the elements `path` giving the path pattern that was matched, `object` giving the object associated with the path, and `params` being a named list of the path parameters from the match

`remove_path(path)`:

Remove a path from the router. Due to the underlying implementation this causes a complete rebuild of the router

Arguments:

- `path`: A string giving the path to remove

Returns:
The object, invisibly

`has_path(path)`:

Check if a given path is present in the router

Arguments:

- `path`: The path pattern to check for

Returns:

A boolean indicating the existence of `path`

`paths()`:

Provides a named list of all the objects, named by their path pattern

Examples

```
# Adapted from path-tree docs
router <- signpost()

router$add_path("/", 1)
router$add_path("/login", 2)
router$add_path("/signup", 3)
router$add_path("/settings", 4)
router$add_path("/settings/:page", 5)
router$add_path("/:user", 6)
router$add_path("/:user/:repo", 7)
router$add_path("/public/:any*", 8)
router$add_path("/:org/:repo/releases/download/:tag/:filename.:ext", 9)
router$add_path("/:org/:repo/tags/:day-:month-:year", 10)
router$add_path("/:org/:repo/actions/:name\\::verb", 11)
router$add_path("/:org/:repo/:page", 12)
router$add_path("/:org/:repo/*", 13)
router$add_path("/api/+", 14)

router$find_object("/")
router$find_object("/login")
router$find_object("/settings/admin")
router$find_object("/viz-rs")
router$find_object("/viz-rs/path-tree")
router$find_object("/rust-lang/rust-analyzer/tags/2022-09-12")
router$find_object("/rust-lang/rust-analyzer/actions/ci:bench")
router$find_object("/rust-lang/rust-analyzer/stargazers")
router$find_object("/rust-lang/rust-analyzer/stargazers/404")
router$find_object("/public/js/main.js")
router$find_object("/api/v1")
```

Index