Package: winch (via r-universe)

October 15, 2026

Title Portable Native and Joint Stack Traces

Version 0.1.3

Date 2026-02-14

Description Obtain the native stack trace and fuse it with R's stack trace for easier debugging of R packages with native code.

License GPL-3

URL https://r-prof.github.io/winch/, https://github.com/r-prof/winch

BugReports https://github.com/r-prof/winch/issues

Imports lifecycle, procmaps (>= 0.0.2)

Suggests DBI, knitr, magrittr, purrr, rlang (>= 0.4.8), rmarkdown, RSQLite, testthat (>= 3.0.0), vctrs

VignetteBuilder knitr

Encoding UTF-8

Biarch yes

RoxygenNote 7.3.3.9000

Config/testthat/edition 3

NeedsCompilation yes

Author Kirill Müller [aut, cre] (ORCID: <https://orcid.org/0000-0002-1416-3412>), R Consortium [fnd], Ian Lance Taylor [aut] (Bundled libbacktrace library), Free Software Foundation [cph] (Bundled libbacktrace library)

Maintainer Kirill Müller <kirill@cynkra.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-02-14 06:40:02 UTC

RemoteUrl https://github.com/cran/winch

RemoteRef HEAD

RemoteSha f576a8bf10eef76cc85430fda0cb56264e1c828f

Contents

winch_available 2
winch_call 2
winch_init_library 3
winch_stop 4
winch_trace_back 4

Index **6**

`winch_available` *Are native tracebacks available?*

Description

Returns `TRUE` if `winch_trace_back()` is supported on this platform.

Usage

```
winch_available()
```

Value

A scalar logical.

Examples

```
winch_available()
```

`winch_call` *Call an R function from native code*

Description

Primarily intended for testing.

Usage

```
winch_call(fun, env = parent.frame())
```

Arguments

`fun`	A function callable without arguments.
`env`	The environment in which to evaluate the function call.

Value

The return value of `fun()`.

See Also

winch_stop()

Examples

```
foo <- function() {
  winch_call(bar)
}

bar <- function() {
  writeLines("Hi!")
}

foo()
```

`winch_init_library` *Set library to collect symbols for native stack traces*

Description

On Windows, function names in native stack traces can be obtained for only one library at a time. Call this function to set the library for which to obtain symbols.

Usage

```
winch_init_library(path = NULL, force = FALSE)
```

Arguments

`path`	Path to the DLL.
`force`	Reinitialize even if the path to the DLL is unchanged from the last call.

Value

This function is called for its side effects.

See Also

winch_call()

Examples

```
winch_init_library(getLoadedDLLs()[["rlang"]][["path"]])
```

winch_stop *Raise an error from native code*

Description

Primarily intended for testing.

Usage

```
winch_stop(message)
```

Arguments

message	The error message.

Value

This function throws an error and does not return.

See Also

winch_call()

Examples

```
try(winch_stop("Test"))
```

winch_trace_back *Native stack trace*

Description

This function returns the native stack trace as a data frame. Each native stack frame corresponds to one row in the returned data frame. Deep function calls come first, the last row corresponds to the running process's entry point.

Usage

```
winch_trace_back()
```

Details

On Windows, call winch_init_library() to return function names for a specific package.

Value

A data frame with the columns:

- `func`: function name
- `ip`: instruction pointer
- `pathname`: path to shared library
- `is_libr`: a logical, `TRUE` if this entry is from R's shared library, determined via `procmaps::path_is_libr()` on the `pathname` component

See Also

`sys.calls()` for the R equivalent.

Examples

```
winch_trace_back()

foo <- function() {
  winch_call(bar)
}

bar <- function() {
  winch_trace_back()
}

foo()
```

Index

procmaps::path_is_libr(), *5*

sys.calls(), *5*

winch_available, 2
winch_call, 2
winch_call(), *3*, *4*
winch_init_library, 3
winch_init_library(), *4*
winch_stop, 4
winch_stop(), *3*
winch_trace_back, 4
winch_trace_back(), *2*