Package ‘winch’

April 20, 2023

Title Portable Native and Joint Stack Traces
Version 0.1.0
Date 2023-04-19
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


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 7.2.3

Config/testthat/edition 3

NeedsCompilation yes

Author Kirill Müller [aut, cre] (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 CRAN

Date/Publication 2023-04-20 07:00:02 UTC

R topics documented:

  winch_available ............................................................ 2
  winch_call ................................................................. 2
Description

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

Usage

```
winch_available()
```

Value

A scalar logical.

Examples

```
winch_available()
```

---

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

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

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

foo()
```

---

**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**

```r
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**

```r
winch_init_library(getLoadedDLLs()[["rlang"]][["path"]])
```
**winch_stop**

*Raise an error from native code*

**Description**

Primarily intended for testing.

**Usage**

```
winch_stop(message)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>The error message.</td>
</tr>
</tbody>
</table>

**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

```r
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(), 2
winch_trace_back, 4
winch_trace_back(), 2