Package ‘zigimports’

October 27, 2026

Type Package

Title Call 'Zig' Code from R

Version 0.1.0

Description Loads shared libraries compiled from 'Zig' ('.so' files on Linux, '.dll' files on Windows) and binds their exported functions to R functions. Arguments are passed by value using the native calling convention, so functions with mixed integer, double, logical, string and vector arguments and with return values can be called. Also compiles and runs 'Zig' code directly from R. Compiling code requires the 'Zig' compiler.

URL https://github.com/lokop5116/zigimports

BugReports https://github.com/lokop5116/zigimports/issues

License GPL-3

Encoding UTF-8

Depends R (>= 3.5.0)

SystemRequirements 'Zig' compiler (>= 0.11), only needed for zig_call() and to build libraries

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

RoxygenNote 7.3.2

NeedsCompilation yes

Author Lakshya Sharma [aut, cre], Amrit Pal [aut]

Maintainer Lakshya Sharma <lakshya6115@gmail.com>

Repository CRAN

Date/Publication 2026-10-27 15:20:02 UTC

Contents

zigimports-package 2
zig_bind 2
zig_call 4
zig_load 6

Index **7**

`zigimports-package` *zigimports: Call 'Zig' Code from R*

Description

Load shared libraries compiled from 'Zig' (`.so` on Linux, `.dll` on Windows) with `zig_load`, bind exported functions with `zig_bind`, and compile and run 'Zig' code from R with `zig_call`.

Author(s)

Maintainer: Firstname Lastname <author@example.com>

Authors:

- Coauthor Name

`zig_bind` *Bind an exported Zig function to an R function*

Description

Looks up a function exported from a library loaded with `zig_load` and returns an ordinary R function that calls it. Arguments are passed by value with the native C calling convention, so functions with mixed argument types, such as `export fn add(a: i32, b: f64) f64`, work as expected.

Usage

```
zig_bind(lib, name, args = NULL, returns = "void")
```

Arguments

`lib`	A `zig_lib` object returned by `zig_load`.
`name`	Name of the exported Zig function.
`args`	Character vector with the type of each argument, or `NULL` for a function without arguments.
`returns`	Return type of the Zig function (default `"void"`).

Details

Supported types (a Zig-style or R-style name may be used):

Type	Zig	R value
`"i32"`	`i32`, `c_int`	single integer
`"f64"`	`f64`	single double
`"bool"`	`bool`	single logical
`"str"`	`[*:0]const u8`	single string
`"ptr_f64"`	`[*]f64`	double vector
`"ptr_i32"`	`[*]i32`	integer or logical vector

`"void"` is also valid as a return type. Functions must be declared with `export fn`, take at most six arguments, and cannot return pointers other than strings. Vector arguments are copied before the call, so the caller's data is never modified; the modified copies are returned (see Value). `NA` integers are passed as the C value `INT_MIN`. The Zig code runs inside the R process: a crash in Zig crashes R.

Value

A function taking the Zig function's arguments. Calling it returns the Zig return value. If the function has vector arguments, their (possibly modified) copies are returned as a named list: in the `"args"` attribute of the return value, or as the result itself for a `void` function.

See Also

zig_load, zig_call

Examples

```
if (nzchar(Sys.which("zig"))) {
  q <- function(p) shQuote(p, type = if (.Platform$OS.type == "windows") "cmd" else "sh")
  src <- system.file("examples", "mathlib.zig", package = "zigimports")
  tmp <- tempfile("zigimports_ex_")
  dir.create(tmp)
  out <- file.path(tmp, paste0("mathlib", .Platform$dynlib.ext))
  status <- system2("zig", c("build-lib", q(src), "-dynamic", "-O", "ReleaseFast",
                             paste0("-femit-bin=", q(out)),
                             "--cache-dir", q(file.path(tmp, "c")),
                             "--global-cache-dir", q(file.path(tmp, "g"))),
                    stdout = FALSE, stderr = FALSE)
  if (status == 0L && file.exists(out)) {
    lib <- zig_load(out)
    add <- zig_bind(lib, "add", args = c("i32", "f64"), returns = "f64")
    add(2L, 0.5)
    total <- zig_bind(lib, "total", args = c("ptr_f64", "i32"), returns = "f64")
    total(c(1, 2, 3), 3L)
    dyn.unload(lib$path)
  }
  unlink(tmp, recursive = TRUE)
}
```

`zig_call` *Compile and run Zig code from R*

Description

Compiles 'Zig' source into a temporary shared library with the `zig` compiler, loads it, calls it, and cleans up. Requires Zig (0.11 or later) to be installed and on the `PATH`, or located through `options(zigimports.zig = ...)` or the `ZIGIMPORTS_ZIG` environment variable. All build files are written to a temporary directory.

Usage

```
zig_call(
  code,
  ...,
  fn = NULL,
  args = NULL,
  returns = "void",
  optimize = c("ReleaseFast", "ReleaseSafe", "ReleaseSmall", "Debug"),
  flags = character(),
  zig = NULL,
  keep = FALSE,
  verbose = FALSE
)
```

Arguments

`code`	Zig source code (character string), or path to a `.zig` file.
`...`	Arguments passed to the Zig function. Must be named in snippet mode.
`fn`	Name of the exported function to call, or `NULL` for snippet mode.
`args`	Argument types (function mode only); see `zig_bind`.
`returns`	Return type of the function or snippet.
`optimize`	Zig optimisation mode.
`flags`	Extra command-line flags for `zig build-lib`, e.g. `"-lc"`.
`zig`	Path or name of the Zig compiler. Defaults to `getOption("zigimports.zig")`, then `ZIGIMPORTS_ZIG`, then `"zig"`.
`keep`	If `TRUE`, keep the temporary build directory and report its location.
`verbose`	If `TRUE`, show the generated source and compiler command.

Details

There are two modes:

- **Function mode** (`fn` given): `code` is a complete Zig source (or the path to a `.zig` file) that defines `export fn <fn>`. The function is called with `...`. Argument types are given by `args`; if `NULL` they are inferred (length-1 double, integer, logical and string values are passed by value, longer vectors as pointers). Use `args` when a length-1 vector must be passed as a pointer.
- **Snippet mode** (`fn = NULL`): `code` is just a function body. Each named argument in `...` becomes a Zig parameter with that name: a pointer for numeric, integer and logical vectors (`[*]f64` or `[*]i32`) and `[*:0]const u8` for a single string. The vector lengths are available as `zig_lens[i]` in argument order, and `std` is already imported. The body can `return` a value if `returns` is set.

See zig_bind for the supported types and conventions.

Value

As for functions created by zig_bind: the Zig return value (with the modified vector arguments in the `"args"` attribute), or, for `returns = "void"`, a named list of the modified vector arguments (invisible `NULL` if there are none).

See Also

zig_load, zig_bind

Examples

```
if (nzchar(Sys.which("zig"))) {
  # Function mode: mixed argument types and a return value
  src <- "export fn add(a: i32, b: f64) f64 {
    return @as(f64, @floatFromInt(a)) + b;
  }"
  zig_call(src, fn = "add", args = c("i32", "f64"), returns = "f64",
           2L, 0.5)

}


if (nzchar(Sys.which("zig"))) {
  # Snippet mode: double every element of a vector
  zig_call("
    var i: usize = 0;
    while (i < @as(usize, @intCast(zig_lens[0]))) : (i += 1) x[i] *= 2;
  ", x = c(1, 2, 3))
}
```

`zig_load` *Load a Zig shared library*

Description

Loads a shared library produced by the 'Zig' compiler (a `.so` file on Linux or a `.dll` file on Windows) so that its exported functions can be bound with `zig_bind`.

Usage

```
zig_load(path)
```

Arguments

`path`	Path to the compiled shared library.

Value

An object of class `"zig_lib"`: a list with the normalised `path`, the library `name`, and the `dll` handle returned by `dyn.load`.

See Also

`zig_bind`, `zig_call`

Examples

```
if (nzchar(Sys.which("zig"))) {
  q <- function(p) shQuote(p, type = if (.Platform$OS.type == "windows") "cmd" else "sh")
  src <- system.file("examples", "mathlib.zig", package = "zigimports")
  tmp <- tempfile("zigimports_ex_")
  dir.create(tmp)
  out <- file.path(tmp, paste0("mathlib", .Platform$dynlib.ext))
  status <- system2("zig", c("build-lib", q(src), "-dynamic", "-O", "ReleaseFast",
                             paste0("-femit-bin=", q(out)),
                             "--cache-dir", q(file.path(tmp, "c")),
                             "--global-cache-dir", q(file.path(tmp, "g"))),
                    stdout = FALSE, stderr = FALSE)
  if (status == 0L && file.exists(out)) {
    lib <- zig_load(out)
    print(lib)
    dyn.unload(lib$path)
  }
  unlink(tmp, recursive = TRUE)
}
```

Index

dyn.load, *6*

zig_bind, *2*, 2, *4–6*
zig_call, *2*, *3*, 4, *6*
zig_load, *2*, *3*, *5*, 6
zigimports *(*zigimports-package*)*, 2
zigimports-package, 2