Package 'cargo'

December 21, 2022

Title Develop R Packages using Rust

Version 0.4.2

Description A framework is provided to develop R packages using 'Rust' <https://www.rust-lang.org/> with minimal overhead, and more wrappers are easily added. Help is provided to use 'Cargo' <https://doc.rust-lang.org/cargo/> in a manner consistent with CRAN policies. 'Rust' code can also be embedded directly in an R script. The package is not official, affiliated with, nor endorsed by the Rust project.

URL https://github.com/dbdahl/cargo-framework (repository)

BugReports https://github.com/dbdahl/cargo-framework/issues

License MIT + file LICENSE | Apache License 2.0

Depends R (>= 4.2.0)

Suggests roxygen2 (>= 7.2.3)

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation no

Author David B. Dahl [aut, cre] (<https://orcid.org/0000-0002-8173-1547>)

Maintainer David B. Dahl <dahl@stat.byu.edu>

Repository CRAN

Date/Publication 2022-12-21 00:40:06 UTC

R topics documented:

api_documentation *Browse API Documentation*

Description

This function opens in a web browser the documentation of the API for the Cargo Framework.

Usage

```
api_documentation(pkgroot = ".")
```

Arguments

pkgroot	The root directory of the package.

Value

NULL, invisibly.

install *Install Rust Toolchain*

Description

This function downloads the 'rustup' installer, run it, and adds targets to compile for all the CRAN build machines.

Usage

```
install(force = FALSE)
```

Arguments

force	If TRUE, installation proceeds without asking for user confirmation.

Value

Invisibly, TRUE if successful and FALSE otherwise.

new_package *Make a Skeleton for a New Package*

Description

A new Rust-based package using the cargo framework is created at the supplied path and the package is installed.

Usage

```
new_package(path, ...)
```

Arguments

path	A path where the package is created. The name of the package is taken as the last element in the file path.
...	Extra arguments that are currently ignored.

prebuild *Prepare for Building the Package Source*

Description

This function provides many tools to be used before building an R package based on the Cargo Framework.

Usage

```
prebuild(
  what = c("register_calls", "document", "vendor", "authors", "all")[5],
  pkgroot = "."
)
```

Arguments

what	A character vector indicating the desired action.
pkgroot	The root directory of the package.

Details

If a package's usage of `base::.Call()` changes, rerun this function with `what="register_calls"` to update the `src/rust/src/registration.rs` file. If you update the `roxygen2` documentation, rerun this function with `what="document"` to update the `*.Rd` files. If a package's Rust code changes a dependency, rerun this function with `what=c("authors","vendor")` to update the files `src/rust/vendor.tar.xz` and to generate the file `authors-scratch.txt` (which will need to be manually incorporated into the `DESCRIPTION` file and then deleted). To perform all of these actions, use `what="all"`.

Value

NULL, invisibly.

---

run *Run Cargo*

Description

This function runs Cargo (Rust's package manager) with the ... arguments passed as command line arguments.

Usage

```
run(
  ...,
  minimum_version = ".",
  search_methods = c("cache", "convention", "path"),
  leave_no_trace = FALSE,
  environment_variables = list(),
  rustflags = NULL,
  verbose = TRUE,
  stdout = "",
  stderr = ""
)
```

Arguments

...	Character vector of command line arguments passed to the cargo command.
minimum_version	A character string representing the minimum version of Rust that is needed. Or a path to the root of a package (i.e., the directory containing the DESCRIPTION file), in which case the value is found from the field: SystemRequirements: Cargo (>= XXXX). For the search_methods being "cache", the shell command rustup is used to upgrade the Cargo installation if needed.
search_methods	A character vector potentially containing values "path", "convention", and "cache". This indicates the methods to use (and their order) when searching for a suitable Cargo installation. "path" indicates to try to use base::Sys.which(). "convention" indicates to try to use the directories .cargo in the user's home directory. "cache" indicates to try to use the directory from the cargo package's own installation as given by the tools::R_user_dir('cargo', 'cache').
leave_no_trace	If TRUE, the CARGO_HOME environment variable is set to a temporary directory that is subsequently deleted.
environment_variables	A named character vector providing environment variables which should be temporarily set while running Cargo. Note that the CARGO_HOME and RUSTUP_HOME environment variables are automatically set when using the "cache" search

method. Also, the CARGO_HOME environment variable is also set when leave_no_trace == TRUE.

rustflags — A character vector from which the CARGO_ENCODED_RUSTFLAGS environment variables is constructed and then temporarily set. Or, if NULL, this environment variable is left unchanged.

verbose — If TRUE, details of the search for Cargo are shown. If FALSE, no details are shown. If it is a connection, then details are shown and also written to the connection.

stdout — See argument of the same name in base::system2().

stderr — See argument of the same name in base::system2().

Value

The same value and behavior as the base::system2() function, except a non-zero exit code will be given in Cargo is not found.

Examples

```
if ( run("--version") != 0 ) {
   message("Cargo is not installed. Please run cargo::install() in an interactive session.")
}
```

---

rust_fn *Define an R Function Implemented in Rust*

Description

This function takes Rust code as a string from the last unnamed argument, takes variable names for all other unnamed arguments, compiles the Rust function, and wraps it as an R function.

Usage

```
rust_fn(
  ...,
  dependencies = character(0),
  minimum_version = "1.31.0",
  verbose = FALSE,
  cached = TRUE,
  longjmp = TRUE,
  invisible = FALSE,
  force = FALSE
)
```

Arguments

`...`	Rust code is taken as a string from the last unnamed argument, and variable names come for all other unnamed arguments. See example.
`dependencies`	A character vector of crate dependencies, e.g., `c('rand = "0.8.5"','rand_pcg = "0.3.1"')`.
`minimum_version`	A character string representing the minimum version of Rust that is needed. Or a path to the root of a package (i.e., the directory containing the DESCRIPTION file), in which case the value is found from the field: `SystemRequirements: Cargo (>= XXXX)`. For the `search_methods` being `"cache"`, the shell command `rustup` is used to upgrade the Cargo installation if needed.
`verbose`	If `TRUE`, Cargo prints compilation details. If `FALSE`, Cargo is run in quiet mode, except for the first time this function is run. If `"never"`, Cargo is always run in quiet mode. In any case, errors in code are always shown.
`cached`	Should Cargo use previously compiled artifacts?
`longjmp`	Should the compiled function use the faster (but experimental) longjmp functionality when Rust code panics?
`invisible`	Should the compiled function return values invisibly?
`force`	If `TRUE`, write to cache directory on first usage without asking for user confirmation.

Value

An R function implemented with the supplied Rust code.

Index