

Package: doParabar (via r-universe)

January 14, 2025

Title 'foreach' Parallel Adapter for 'parabar' Backends

Version 1.0.2

Description Provides a 'foreach' parallel adapter for 'parabar' backends. This package offers a minimal implementation of the '%dopar%' operator, enabling users to run 'foreach' loops in parallel, leveraging the parallel and progress-tracking capabilities of the 'parabar' package. Learn more about 'parabar' and 'doParabar' at <https://parabar.mihaiconstantin.com>.

License MIT + file LICENSE

URL <https://github.com/mihaiconstantin/doParabar>,
<https://parabar.mihaiconstantin.com/articles/foreach>

BugReports <https://github.com/mihaiconstantin/doParabar/issues>

Imports parabar, foreach, iterators, utils

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Collate 'doPar.R' 'doParabar-package.R' 'helpers.R' 'logo.R'
'registerDoParabar.R'

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Repository <https://mihaiconstantin.r-universe.dev>

RemoteUrl <https://github.com/mihaiconstantin/doparabar>

RemoteRef HEAD

RemoteSha 332e5017d43cde3673030ea63b44859b9537c66b

Contents

registerDoParabar	2
Index	4

registerDoParabar *Register Parallel Implementation*

Description

The `registerDoParabar()` function registers the provided `backend` created by `parabar::start_backend()` to be used as the parallel processing backend for the `foreach::%dopar%` operator implementation.

Usage

```
registerDoParabar(backend)
```

Arguments

`backend` An object of class `parabar::Backend` representing the backend to be used for the `foreach::%dopar%` operator implementation.

Details

Additional information about the registered parallel backend can be extracted using the `foreach::getDoParName()`, `foreach::getDoParRegistered()`, `foreach::getDoParVersion()`, and `foreach::getDoParWorkers()` functions. See the **Examples** section.

Value

The `registerDoParabar()` function returns void.

Completeness

The parallel backend implementation for the `foreach::%dopar%` operator is provided by the `doPar()` function. Please check the **Details** section of its documentation to understand the extent of completeness of the implementation.

See Also

`doParabar`, `doPar()`, `parabar::start_backend()` and `parabar::stop_backend()`.

Examples

```
# Manually load the libraries.
library(doParabar)
library(parabar)
library(foreach)

# Create an asynchronous `parabar` backend.
backend <- start_backend(cores = 2, cluster_type = "psock", backend_type = "async")

# Register the backend with the `foreach` package for the `%dopar%` operator.
registerDoParabar(backend)
```

```
# Get the parallel backend name.
getDoParName()

# Check that the parallel backend has been registered.
getDoParRegistered()

# Get the current version of backend registration.
getDoParVersion()

# Get the number of cores used by the backend.
getDoParWorkers()

# Define some variables strangers to the backend.
x <- 10
y <- 100
z <- "Not to be exported."

# Used the registered backend to run a task in parallel via `foreach`.
results <- foreach(i = 1:300, .export = c("x", "y"), .combine = c) %dopar% {
  # Sleep a bit.
  Sys.sleep(0.01)

  # Compute and return.
  i + x + y
}

# Show a few results.
head(results, n = 10)
tail(results, n = 10)

# Verify that the variable `z` was not exported.
try(evaluate(backend, z))

# To make packages available on the backend, see the `.packages` argument.

# Stop the backend.
stop_backend(backend)
```

Index

backend, [2](#)

doPar(), [2](#)

doParabar, [2](#)

foreach::%dopar%, [2](#)

foreach::getDoParName(), [2](#)

foreach::getDoParRegistered(), [2](#)

foreach::getDoParVersion(), [2](#)

foreach::getDoParWorkers(), [2](#)

parabar::Backend, [2](#)

parabar::start_backend(), [2](#)

parabar::stop_backend(), [2](#)

registerDoParabar, [2](#)

registerDoParabar(), [2](#)