Package: orderly.server (via r-universe)

October 5, 2024

Title Serve Orderly

Version 0.3.32

Description Run orderly reports as a server.

License MIT + file LICENSE

Author Rich FitzJohn

Maintainer Rich FitzJohn <rich.fitzjohn@gmail.com>

Additional_repositories https://mrc-ide.github.io/drat

Imports DBI, R6, docopt (>= 0.7.1), ids, jsonlite, lgr, orderly (>= 1.4.10), porcelain (>= 0.1.9), processx, redux, rrq (== 0.6.0), utils, yaml

Suggests callr, gert (>= 1.6.0), httr, jsonvalidate (>= 1.2.2), mockery, RSQLite, testthat, withr, zip

RoxygenNote 7.2.0

Encoding UTF-8

Remotes mrc-ide/rrq@ebca6d1, reside-ic/porcelain, ropensci/jsonvalidate, vimc/orderly

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

RemoteUrl https://github.com/vimc/orderly.server

RemoteRef master

RemoteSha bcaba2ef63f05736f51043540289d8957bb2df7f

Contents

`orderly_runner` *Orderly runner*

Description

An orderly runner. This is used to run reports as a server process. It's designed to be used in conjunction with OrderlyWeb, so there is no "draft" stage and reports are committed as soon as they are run. This function is not intended for human end users, only for creating automated tools for use with orderly.

Usage

```
orderly_runner(path, identity = NULL, queue_id = NULL, workers = 1)
```

Arguments

`path` Path to use

`identity` The name of the server identity, as listed in orderly_config.yml's remote section. If not given, then we will look to see if the orderly configuration disallows branch changes (based on the `ORDERLY_API_SERVER_IDENTITY` environment variable. Used to set configuration specific to this server (e.g., host, port, teams notification URL, default branch name, etc).

`queue_id` ID of an existing queue to connect to, creates a new one if NULL

`workers` Number of workers to spawn

Value

A runner object, with methods designed for internal use only.

Examples

```
available <- redux::redis_available()
if (available) {
  path <- orderly:::prepare_orderly_git_example()
  runner <- orderly.server::orderly_runner(path[["local"]], workers = 0)
}
```

server *Run orderly server*

Description

Run orderly server

Usage

```
server(
  path,
  port,
  host = "0.0.0.0",
  go_signal = NULL,
  queue_id = NULL,
  workers = 1,
  backup_period = 600,
  timeout_rate_limit = 2 * 60,
  log_level = "info",
  identity = NULL
)
```

Arguments

`path` Path to serve

`port` Port to serve on

`host` Optional

`go_signal` If given, we poll for a file `go_signal` (within `path`) before starting. This is designed primarily for use with docker where the data volume may not be ready at the same time as the process container (and indeed won't be if the container is used to provision the volume). During this period the server will not respond to any http requests.

`queue_id` ID of an existing queue to connect to, creates a new one if NULL

`workers` Number of workers to spawn

`backup_period` How frequently should backup be run, if NULL backup is skipped

`timeout_rate_limit` How frequently should the API check for timeouts default 2 mins.

`log_level` The "lgr" log level to use

`identity` Optional server identity

Index