Package: batchr (via r-universe)

November 1, 2024

Title Batch Process Files

Version 0.0.2.9000

Description Processes multiple files with a user-supplied function. The key design principle is that only files which were last modified before the directory was configured are processed. A hidden file stores the configuration time and function etc while successfully processed files are automatically touched to update their modification date. As a result batch processing can be stopped and restarted and any files created (or modified or deleted) during processing are ignored.

License MIT + file LICENSE

URL https://poissonconsulting.github.io/batchr/

BugReports https://github.com/poissonconsulting/batchr/issues

Depends R (>= 3.6)

Imports chk, cli, furrr, hms, hmstimer, parallel, stats, yesno

Suggests covr, future, knitr, rmarkdown, testthat, tibble, withr

VignetteBuilder knitr

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

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

RemoteUrl https://github.com/poissonconsulting/batchr

RemoteRef HEAD

RemoteSha 47741c08466fc7124723d2bc264ab1e3b9faa9c6

Contents

`batch_cleanup` *Cleanup Batch Processing*

Description

Deletes configuration file created by `batch_config()` and log file created by `batch_run()`.

Usage

```
batch_cleanup(
  path,
  force = FALSE,
  remaining = FALSE,
  failed = NA,
  recursive = FALSE,
  silent = FALSE
)
```

Arguments

`path`	A string of the path to the directory with the files for processing.
`force`	A flag specifying whether to delete configuration and log files even if there are files remaining to be processed.
`remaining`	A flag specifying whether to delete any files that are remaining to be processed (only applied when `force = TRUE`). Files that have been processed are never deleted.
`failed`	A logical scalar specifying how to treat files that previously failed to process. If FALSE (the default) failed files are excluded, if NA they are included and if TRUE they are only included.

recursive	A flag specifying whether to recurse into subdirectories when cleaning up. This is unrelated to the recurse option of batch_config() and is only expected to be used if the user has neglected to clean up multiple nested directories.
silent	A flag specifying whether to suppress warnings (and messages).

Details

The batch_completed() function can be used to test if batch processing is complete.

Value

A named logical vector indicating which directories were successfully cleaned up.

See Also

batch_process()

Examples

```
path <- tempdir()
write.csv(mtcars, file.path(path, "file1.csv"))
batch_config(function(x) TRUE, path, regexp = "[.]csv$")
batch_run(path, ask = FALSE)
batch_cleanup(path)
unlink(file.path(path, "file1.csv"))
```

---

batch_completed *Batch Completed?*

Description

Tests if there are any remaining files to process as listed by batch_files_remaining().

Usage

```
batch_completed(path, failed = FALSE)
```

Arguments

path	A string of the path to the directory with the files for processing.
failed	A logical scalar specifying how to treat files that previously failed to process. If FALSE (the default) failed files are excluded, if NA they are included and if TRUE they are only included.

Details

By default, files that previously failed to process are excluded.

Value

A flag specifying whether batch processing is complete.

See Also

batch_process()

Examples

```
path <- tempdir()
write.csv(mtcars, file.path(path, "file1.csv"))
batch_config(function(x) TRUE, path, regexp = "[.]csv$")
batch_completed(path)
batch_run(path, ask = FALSE)
batch_completed(path)
batch_cleanup(path)
unlink(file.path(path, "file1.csv"))
```

---

batch_config *Configure Batch Processing*

Description

Configures a directory for batch file processing by batch_run().

Usage

```
batch_config(fun, path, regexp = ".*", recurse = FALSE, ...)
```

Arguments

fun	A function to process each of the files. fun's first argument should be a string of the path to a single file. If processing is unsuccessful fun should return FALSE or throw an error (error messages are caught and automatically logged). If fun deletes or modifies the file then it is no longer considered for processing.
path	A string of the path to the directory with the files for processing.
regexp	A string of a regular expression. Only non-hidden file names which match the regular expression will be batch processed.
recurse	A flag specifying whether to recurse into path's subdirectories.
...	Additional arguments passed to fun.

Details

batch_config() creates a hidden configuration file in path named '.batchr.rds'.

The contents of the file can be read using batch_config_read() or updated using batch_reconfig_fun().

Configuration is only possible if the directory does not already contain a configuration file. If recurse = TRUE then the subdirectories must also not contain configuration files.

The regexp must match at least one non-hidden file in the directory or if recurse = TRUE in the directory or subdirectories. Hidden files are excluded to prevent accidental modification of system files.

Value

An invisible character vector of the paths to the files to that will be processed when batch_run() is called.

See Also

batch_process() and batch_run()

Examples

```
path <- tempdir()
write.csv(mtcars, file.path(path, "file1.csv"))
batch_config(function(x) TRUE, path, regexp = "[.]csv$")
batch_run(path, ask = FALSE)
batch_cleanup(path)
unlink(file.path(path, "file1.csv"))
```

---

batch_config_read *Read Configuration File*

Description

Reads the values in the configuration file created by batch_config().

Usage

```
batch_config_read(path)
```

Arguments

path	A string of the path to the directory with the files for processing.

Value

A named list of the configuration values.

See Also

batch_process() and batch_log_read()

Examples

```
path <- tempdir()
write.csv(mtcars, file.path(path, "file1.csv"))
batch_config(function(x) TRUE, path, regexp = "[.]csv$")
batch_config_read(path)
batch_cleanup(path, force = TRUE, remaining = TRUE)
unlink(file.path(path, "file1.csv"))
```

---

batch_files_remaining *Batch Files*

Description

Gets the names of the files that are remaining to be processed by batch_run().

Usage

```
batch_files_remaining(path, failed = FALSE)
```

Arguments

path	A string of the path to the directory with the files for processing.
failed	A logical scalar specifying how to treat files that previously failed to process. If FALSE (the default) failed files are excluded, if NA they are included and if TRUE they are only included.

Details

batch_completed() can be used to test if there are any files remaining.

Value

A character vector of the names of the remaining files.

See Also

batch_process() and batch_run()

Examples

```
path <- tempdir()
write.csv(mtcars, file.path(path, "file1.csv"))
batch_config(function(x) TRUE, path, regexp = "[.]csv$")
batch_files_remaining(path)
batch_run(path, ask = FALSE)
batch_files_remaining(path)
batch_cleanup(path)
unlink(file.path(path, "file1.csv"))
```

---

batch_file_status *Batch File Status*

Description

Gets the current status (SUCCESS, FAILURE, REMAING) of each eligible file in path.

Usage

```
batch_file_status(path)
```

Arguments

path	A string of the path to the directory with the files for processing.

Value

A tibble with four columns:

type A character vector indicating SUCCESS, FAILURE or REMAING

time A hms vector of the file processing time

file A character vector of the file name

error A character vector of the error message (or NA if no error)

See Also

batch_log_read()

Examples

```
path <- tempdir()
write.csv(mtcars, file.path(path, "file1.csv"))
batch_config(function(x) TRUE, path, regexp = "[.]csv$")
batch_file_status(path)
batch_run(path, ask = FALSE)
batch_file_status(path)
batch_cleanup(path)
unlink(file.path(path, "file1.csv"))
```

batch_is_clean *Is Clean*

Description

Tests whether directory contains configuration file created by `batch_config()`.

Usage

```
batch_is_clean(path, recurse = FALSE)
```

Arguments

path	A string of the path to the directory with the files for processing.
recurse	A flag specifying whether to recurse into path's subdirectories.

Value

A flag specifying whether the directory is clean.

See Also

`batch_cleanup()`

Examples

```
path <- tempdir()
batch_is_clean(path)
write.csv(mtcars, file.path(path, "file1.csv"))
batch_config(function(x) TRUE, path, regexp = "[.]csv$")
batch_is_clean(path)
batch_cleanup(path, force = TRUE, remaining = TRUE)
batch_is_clean(path)
unlink(file.path(path, "file1.csv"))
```

batch_log_read *Read Log File*

Description

Reads the values in the log file created by `batch_run()`.

Usage

```
batch_log_read(path)
```

Arguments

path	A string of the path to the directory with the files for processing.

Value

A tibble with four columns:

type A character vector indicating SUCCESS or FAILURE

time A hms vector of the file processing time

file A character vector of the file name

error A character vector of the error message (or NA if no error)

See Also

batch_process() and batch_config_read()

Examples

```
path <- tempdir()
write.csv(mtcars, file.path(path, "file1.csv"))
batch_config(function(x) TRUE, path, regexp = "[.]csv$")
batch_log_read(path)
batch_run(path, ask = FALSE)
batch_log_read(path)
batch_cleanup(path)
unlink(file.path(path, "file1.csv"))
```

---

batch_process *Batch File Processing*

Description

Performs batch processing of files in a directory using the batch_config(), batch_run() and batch_cleanup() functions. For more control the user should call these three functions instead.

Usage

```
batch_process(
  fun,
  path,
  regexp = ".*",
  recurse = FALSE,
  progress = FALSE,
  force = TRUE,
  report = TRUE,
  seeds = NULL,
  options = furrr::furrr_options(),
```

```
  ask = getOption("batchr.ask", TRUE),
  ...
)
```

Arguments

fun	A function to process each of the files. fun's first argument should be a string of the path to a single file. If processing is unsuccessful fun should return FALSE or throw an error (error messages are caught and automatically logged). If fun deletes or modifies the file then it is no longer considered for processing.
path	A string of the path to the directory with the files for processing.
regexp	A string of a regular expression. Only non-hidden file names which match the regular expression will be batch processed.
recurse	A flag specifying whether to recurse into path's subdirectories.
progress	A flag specifying whether to print a progress bar.
force	A flag specifying whether to delete configuration and log files even if there are files remaining to be processed.
report	A flag specifying whether to outputs a report of the status of individual files to the console.
seeds	A named list of the L'Ecuyer-CMRG seed to use for each file. If NULL then seeds is batch_seeds(files).
options	The future specific options to use with the workers. seed must be FALSE.
ask	A flag specifying whether to ask before starting to process the files.
...	Additional arguments passed to fun.

Value

An invisible flag indicating whether all the files where successfully processed.

See Also

batch_config(), batch_run() and batch_cleanup()

Examples

```
path <- tempdir()
write.csv(mtcars, file.path(path, "file1.csv"))
batch_process(function(x) TRUE, path, regexp = "[.]csv$", ask = FALSE)
unlink(file.path(path, "file1.csv"))
```

batch_reconfig_fileset

Reconfigures Batch Processing File Set

Description

Updates the regular expression and/or recurse argument that were provided when a directory was configured (using batch_config()).

Usage

```
batch_reconfig_fileset(path, regexp = NULL, recurse = NULL)
```

Arguments

path	A string of the path to the directory with the files for processing.
regexp	A string of a regular expression. Only non-hidden file names which match the regular expression will be batch processed.
recurse	A flag specifying whether to recurse into path's subdirectories.

Details

batch_reconfig_fileset() is useful for including or excluding particular files.

It should be noted that batch_reconfig_fun() does not alter the configuration time.

In order to process previously failed files batch_run() should be called with failed = NA or failed = TRUE.

Value

An invisible character vector of the paths to the files remaining to be processed.

See Also

batch_process() and batch_config()

Examples

```
path <- tempdir()
write.csv(mtcars, file.path(path, "file1.csv"))
batch_config(function(x) TRUE, path, regexp = "[.]csv$")
batch_config_read(path)
batch_reconfig_fileset(path, regexp = "file\\d+[.]csv$")
batch_config_read(path)
batch_cleanup(path, force = TRUE, remaining = TRUE)
unlink(file.path(path, "file1.csv"))
```

batch_reconfig_fun *Reconfigures Batch Processing Function*

Description

Updates the function and function arguments that were provided when a directory was configured (using batch_config()).

Usage

```
batch_reconfig_fun(path, fun, ...)
```

Arguments

path	A string of the path to the directory with the files for processing.
fun	A function to process each of the files. fun's first argument should be a string of the path to a single file. If processing is unsuccessful fun should return FALSE or throw an error (error messages are caught and automatically logged). If fun deletes or modifies the file then it is no longer considered for processing.
...	Additional arguments passed to fun.

Details

batch_reconfig_fun() is useful if a new version of the function is required to successfully process some of the files.

It should be noted that batch_reconfig_fun() does not alter the configuration time.

In order to process previously failed files batch_run() should be called with failed = NA or failed = TRUE.

Value

An invisible character vector of the paths to the files remaining to be processed.

See Also

batch_process() and batch_config()

batch_report *Batch Report*

Description

Outputs a report of the status of individual files to the console.

Usage

```
batch_report(path)
```

Arguments

path	A string of the path to the directory with the files for processing.

Value

An invisible NULL. The function is called for its side-effect of outputting a report of the status of individual files to the console.

See Also

batch_file_status()

Examples

```
path <- tempdir()
write.csv(mtcars, file.path(path, "file1.csv"))
batch_config(function(x) TRUE, path, regexp = "[.]csv$",)
batch_report(path)
batch_run(path, ask = FALSE)
batch_report(path)
batch_cleanup(path)
unlink(file.path(path, "file1.csv"))
```

batch_run *Runs Batch Processing*

Description

Starts (or restarts if previously stopped) processing the remaining files specified by batch_config().

Usage

```
batch_run(
  path,
  failed = FALSE,
  progress = FALSE,
  files = NULL,
  seeds = NULL,
  options = furrr::furrr_options(),
  ask = getOption("batchr.ask", TRUE)
)
```

Arguments

path	A string of the path to the directory with the files for processing.
failed	A logical scalar specifying how to treat files that previously failed to process. If FALSE (the default) failed files are excluded, if NA they are included and if TRUE they are only included.
progress	A flag specifying whether to print a progress bar.
files	A character vector of the remaining files to process. If NULL then files is batch_files_remaining(path, failed).
seeds	A named list of the L'Ecuyer-CMRG seed to use for each file. If NULL then seeds is batch_seeds(files).
options	The future specific options to use with the workers. seed must be FALSE.
ask	A flag specifying whether to ask before starting to process the files.

Details

batch_run() logs all file processing attempts together with the the type (SUCCESS or FAILURE), the system time in UTC, the file name and any error messages. The hidden log file can be read using batch_log_read().

batch_files_remaining() provides a vector of the files that are remaining to be processed.

When processing is complete the hidden configuration file and hidden log file can be deleted using batch_cleanup().

If a remaining file is removed or modified by a separate process, batch_run() throws an error.

Value

An invisible named logical vector indicating for each file whether it was successfully processed.

See Also

batch_process(), batch_config() and batch_cleanup()

Examples

```
path <- tempdir()
write.csv(mtcars, file.path(path, "file1.csv"))
batch_config(function(x) TRUE, path, regexp = "[.]csv$")
batch_run(path, ask = FALSE)
batch_cleanup(path)
unlink(file.path(path, "file1.csv"))
```

---

batch_seeds *L'Ecuyer-CMRG Seeds*

Description

Generates a named list of L'Ecuyer-CMRG seeds.

Usage

```
batch_seeds(files = batch_files_remaining())
```

Arguments

files	A character vector of the names of the files.

Value

A named list of the L'Ecuyer-CMRG seed for each file name.

Examples

```
batch_seeds(c("a", "b"))
```

Index