Package ‘knitrProgressBar’

July 22, 2025

Type Package

Title Provides Progress Bars in 'knitr'

Version 1.1.1

Description Provides a progress bar similar to 'dplyr' that can write progress out to a variety of locations, including stdout(), stderr(), or from file(). Useful when using 'knitr' or 'rmarkdown',
and you still want to see progress of calculations in the terminal.

BugReports `https://github.com/rmflight/knitrProgressBar/issues`

URL `https://rmflight.github.io/knitrProgressBar/`

License MIT + file LICENSE

Encoding UTF-8

Imports R6, R.oo

Suggests knitr, rmarkdown, purrr, testthat, covr, mockr, withr, parallel

VignetteBuilder knitr

RoxygenNote 7.3.2

NeedsCompilation no

Author Robert M Flight [aut, cre],
Hadley Wickham [ctb] (Author of included dplyr fragments),
Romain Francois [ctb] (Author of included dplyr fragments),
Lionel Henry [ctb] (Author of included dplyr fragments),
Kirill Müller [ctb] (Author of included dplyr fragments),
RStudio [cph] (Copyright holder of included dplyr fragments)

Maintainer Robert M Flight <`rflight79@gmail.com`>

Repository CRAN

Date/Publication 2024-07-23 18:00:01 UTC

Contents

`make_kpb_output_decisions` *Progress Output Location*

Description

Provides functionality to decide **how** the progress should be written, if at all.

Usage

```
make_kpb_output_decisions()
```

Details

This function makes decisions about **how** the progress bar should be displayed based on whether:

1. The code is being run in an interactive session or not
2. The code is part of a `knitr` evaluation using `knit()` or `rmarkdown::render()`
3. Options set by the user. These options include:
   (a) **kpb.suppress_noninteractive**: a logical value. Whether to suppress output when being run non-interactively.
   (b) **kpb.use_logfile**: logical, should a log-file be used for output?
   (c) **kpb.log_file**: character string defining the log-file to use. **kpb.use_logfile** must be `TRUE`.
   (d) **kpb.log_pattern**: character string providing a pattern to use, will be combined with the chunk label to create a log-file for each knitr chunk. **kpb.use_logfile** must be `TRUE`.

Based on these, it will either return a newly opened connection, either via `stderr()`, `stdout()`, or a file connection via `file("logfile.log", open = "w")`. Note that for files this will overwrite a previously existing file, and the contents will be lost.

Value

a write-able connection or NULL

Examples

```
## Not run:
# suppress output when not interactive
options(kpb.suppress_noninteractive = TRUE)

# use a log-file, will default to kpb_output.txt
options(kpb.use_logfile = TRUE)

# use a specific log-file
options(kpb.use_logfile = TRUE)
options(kpb.log_file = "progress.txt")

# use a log-file based on chunk names
options(kpb.use_logfile = TRUE)
options(kpb.log_pattern = "pb_out_")
# for a document with a chunk labeled: "longcalc", this will generate "pb_out_longcalc.log"

## End(Not run)
```

`progress_estimated` *Progress bar with estimated time.*

Description

This provides a reference class representing a text progress bar that displays the estimated time remaining. When finished, it displays the total duration. The automatic progress bar can be disabled by setting `progress_location = NULL`.

Usage

```
progress_estimated(
  n,
  min_time = 0,
  progress_location = make_kpb_output_decisions()
)
```

Arguments

`n`	Total number of items
`min_time`	Progress bar will wait until at least `min_time` seconds have elapsed before displaying any results.
`progress_location`	where to write the progress to. Default is to make decisions based on location type using `make_kpb_output_decisions()`.

Value

A ref class with methods `tick()`, `print()`, `pause()`, and `stop()`.

See Also

make_kpb_output_decisions()

Examples

```
p <- progress_estimated(3)
p$tick()
p$tick()
p$tick()

p <- progress_estimated(3)
for (i in 1:3) p$pause(0.1)$tick()$print()

p <- progress_estimated(3)
p$tick()$print()$
 pause(1)$stop()

# If min_time is set, progress bar not shown until that many
# seconds have elapsed
p <- progress_estimated(3, min_time = 3)
for (i in 1:3) p$pause(0.1)$tick()$print()

## Not run:
p <- progress_estimated(10, min_time = 3)
for (i in 1:10) p$pause(0.5)$tick()$print()

# output to stderr
p <- progress_estimated(10, progress_location = stderr())

# output to a file
p <- progress_estimated(10, progress_location = tempfile(fileext = ".log"))

## End(Not run)
```

---

set_progress_mp *multi process progress indicator*

Description

Sets up a progress object that writes to a shared file to indicate the total progress. Progress can be monitored by watch_progress_mp.

Usage

```
set_progress_mp(write_location = NULL)
```

Arguments

write_location	where to save progress to

Value

ProgressMP

See Also

watch_progress_mp

`update_progress` *updating progress bars*

Description

Takes care of updating a progress bar and stopping when appropriate

Usage

```
update_progress(.pb = NULL)
```

Arguments

`.pb`	the progress bar object

Value

the progress bar

`watch_progress_mp` *watch progress from multi process*

Description

sets up a "watcher" function that will report on the progress of a multi-process process that is being indicated by `set_progress_mp`.

Usage

```
watch_progress_mp(
  n,
  min_time = 0,
  watch_location = NULL,
  progress_location = make_kpb_output_decisions()
)
```

Arguments

`n`	number of times process is running
`min_time`	how long to wait
`watch_location`	where is the progress being written to
`progress_location`	where to write the progress output

Value

ProgressMPWatcher

See Also

set_progress_mp

Index