Package: future.tools (via r-universe)

September 12, 2024

Version 0.1.0-9002

Title Tools for Working with Futures

Depends R (>= 3.3.0), future (>= 1.32.0)

Imports grDevices, dplyr, ggplot2

Suggests future.apply

Description Tools for Working with Futures.

License LGPL (>= 2.1)

LazyLoad TRUE

Encoding UTF-8

URL https://future.tools.futureverse.org,
https://github.com/futureverse/future.tools

BugReports https://github.com/futureverse/future.tools/issues

RoxygenNote 7.3.1

Roxygen list(markdown = TRUE)

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

RemoteUrl https://github.com/futureverse/future.tools

RemoteRef main

RemoteSha 8a7300cc2b7223504d9a965f40318da6c273a06b

Contents

capture_journals *Evaluate an R expression while collecting journals from completed futures*

Description

Evaluate an R expression while collecting journals from completed futures

Usage

```
capture_journals(expr, substitute = TRUE, envir = parent.frame())
```

Arguments

expr	The R expression to evaluate
substitute	If TRUE, then expr is substituted, otherwise not.
envir	The environment where expr should be evaluated

Details

This function evaluates an R expression and capture the journals signaled by futures as they are completed. A future journal comprise a log of events appearing during the life-span of a future, e.g. the timestamps when the future was created, launched, queried, resolved, and its results are collected.

Value

A list of FutureJournal:s.

Examples

```
slow_fcn <- function(x) {
  Sys.sleep(x / 10)
  sqrt(x)
}

plan(multisession, workers = 2)
js <- capture_journals({
  fs <- lapply(3:1, FUN = function(x) future(slow_fcn(x)))
  value(fs)
})

## Summarize all journals
js_all <- Reduce(rbind, js)
print(summary(js_all), digits = 2L)

## Shut down parallel workers
plan(sequential)
```

ggjournal *Create a Future Journal Plot*

Description

Create a Future Journal Plot

Usage

```
ggjournal(
  x,
  style = c("future", "future-worker", "worker"),
  flatten = FALSE,
  time_range = getOption("future.tools.ggjournal.time_range", NULL),
  item_range = getOption("future.tools.ggjournal.item_range", NULL),
  events = NULL,
  baseline = TRUE,
  label_fmt = "#%s",
  annotate = c("future_label"),
  arrows = c("launch", "gather"),
  layer_height = c(1/4, 1/4, 1/4, 1/8),
  ...
)
```

Arguments

x	A list of future::Future or FutureJournal objects.
style	(character string) One of "future", "future-worker", and "worker".
flatten	(logical) If TRUE, futures are not separated vertically.
time_range	(optional vector of length two) The range of time to displayed.
item_range	(optional vector of length two) The range of future or worker indices to displayed.
events	(character vector; optional) Events to be displayed. If NULL, then all events are displayed.
baseline	(POSIXct; optional) A timestamp to server as time zero for the relative timestamps. If TRUE (default), then the earliest timepoint observed is used as the baseline.
label_fmt	(format string; optional) Used to create labels if future_label is missing. If NULL, no labels are created.
annotate	(character vector) Additional annotations to add.
arrows	(character vector) Type of arrows to draw.
layer_height	(integer vector of length four) Height of each of the four possible layers of stacked events. Their total height, the sum, should be less than one in order for futures to not overlap.
...	Currently not used.

Value

A ggplot2::ggplot object.

Examples

```
library(future.apply)
library(future)

slow_fcn <- function(x) {
  Sys.sleep(x / 10)
  sqrt(x)
}

## Plot with fixed x and y limits
ggjournal_x <- function(js) {
  for (style in c("future", "worker")) {
    item_range <- if (style == "future") c(1, 5) else c(0, 1.8)
    print(ggjournal(js, style = style,
                    time_range = c(0, 2.0), item_range = item_range))
  }
}


plan(sequential)

js <- capture_journals({
  fs <- lapply(5:1, FUN = function(x) future(slow_fcn(x)))
  vs <- value(fs)
})
ggjournal_x(js)

js <- capture_journals({
  vs <- future_lapply(5:1, FUN = slow_fcn)
})
ggjournal_x(js)


plan(multisession, workers = 2)

js <- capture_journals({
  fs <- lapply(5:1, FUN = function(x) future(slow_fcn(x)))
  vs <- value(fs)
})
ggjournal_x(js)

js <- capture_journals({
  vs <- future_lapply(5:1, FUN = slow_fcn)
})
ggjournal_x(js)


## Shut down parallel workers
```

```
plan(sequential)
```

Index