Package 'chronicler'

May 17, 2022

Title Add Logging to Functions

Version 0.2.0

Description Decorate functions to make them return enhanced output. The enhanced output consists in an object of type
'chronicle' containing the result of the function applied to its arguments, as well as a log detailing when the function
was run, what were its inputs, what were the errors (if the function failed to run) and other useful information.
Tools to handle decorated functions are included, such as a forward pipe operator that makes chaining decorated functions possible.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.1.2

BugReports https://github.com/b-rodrigues/chronicler/issues

Depends R (>= 4.1),

Imports clipr, diffobj, dplyr, maybe, rlang, stringr, tibble

Suggests knitr, lubridate, purrr, rmarkdown, testthat (>= 3.1.4),
tidyr

VignetteBuilder knitr

Config/testthat/edition 3

LazyData true

NeedsCompilation no

Author Bruno Rodrigues [aut, cre, cph]
(<https://orcid.org/0000-0002-3211-3689>)

Maintainer Bruno Rodrigues <bruno@brodrigues.co>

Repository CRAN

Date/Publication 2022-05-17 09:40:04 UTC

R topics documented:

`as_chronicle` *Coerce an object to a chronicle object.*

Description

Coerce an object to a chronicle object.

Usage

```
as_chronicle(.x, .log_df = data.frame())
```

Arguments

`.x`	Any object.
`.log_df`	Used internally, the user does need to interact with it. Defaults to an empty data frame.

Value

Returns a chronicle object with the object as the $value.

Examples

```
as_chronicle(3)
```

avia — *Air passenger transport between the main airports of Luxembourg and their main partner airports*

Description

A non-tidy dataset from EUROSTAT which can be found here.

Usage

```
avia
```

Format

A data frame with 510 rows and 238 columns.

bind_record — *Evaluate a decorated function; used to chain multiple decorated functions.*

Description

Evaluate a decorated function; used to chain multiple decorated functions.

Usage

```
bind_record(.c, .f, ...)
```

Arguments

.c	A chronicle object.
.f	A chronicle function to apply to the returning value of .c.
...	Further parameters to pass to .f.

Value

A chronicle object.

Examples

```
r_sqrt <- record(sqrt)
r_exp <- record(exp)
3 |> r_sqrt() |> bind_record(r_exp)
```

check_diff *Check the output of the diff column*

Description

Check the output of the diff column

Usage

```
check_diff(.c, columns = c("ops_number", "function"))
```

Arguments

.c	A chronicle object.
columns	Columns to select for the output. Defaults to c("ops_number", "function").

Details

`diff` is an option argument to the `record()` function. When `diff` = "full", a diff of the input and output of the decorated function gets saved, and if `diff` = "summary" only a summary of the diff is saved.

Value

A data.frame with the selected columns and column "diff_obj".

Examples

```
r_subset <- record(subset, diff = "full")
result <- r_subset(mtcars, select = am)
check_diff(result) # <- this is the data frame listing the operations and the accompanying diffs
check_diff(result)$diff_obj # <- actually look at the diffs
```

check_g *Check the output of the .g function*

Description

Check the output of the .g function

Usage

```
check_g(.c, columns = c("ops_number", "function"))
```

Arguments

`.c`	A chronicle object.
`columns`	Columns to select for the output. Defaults to c("ops_number", "function").

Details

`.g` is an option argument to the `record()` function. Providing this optional function allows you, at each step of a pipeline, to monitor interesting characteristics of the `value` object. See the package's Readme file for an example with data frames.

Value

A data.frame with the selected columns and column "g".

Examples

```
r_subset <- record(subset, .g = dim)
result <- r_subset(mtcars, select = am)
check_g(result)
```

`flatten_record` *Flatten nested chronicle objects*

Description

Flatten nested chronicle objects

Usage

```
flatten_record(.c)
```

Arguments

`.c`	A nested chronicle object, where the $value element is itself a chronicle object

Value

Returns `.c` where value is the actual value, and logs are concatenated.

Examples

```
r_sqrt <- record(sqrt)
r_log <- record(log)
a <- as_chronicle(r_log(10))
a
flatten_record(a)
```

fmap_record *Evaluate a non-chronicle function on a chronicle object.*

Description

Evaluate a non-chronicle function on a chronicle object.

Usage

```
fmap_record(.c, .f, ...)
```

Arguments

`.c`	A chronicle object.
`.f`	A non-chronicle function.
`...`	Further parameters to pass to `.f`.

Value

Returns the result of `.f(.c$value)` as a new chronicle object.

Examples

```
as_chronicle(3) |> fmap_record(sqrt)
```

is_chronicle *Checks whether an object is of class "chronicle"*

Description

Checks whether an object is of class "chronicle"

Usage

```
is_chronicle(.x)
```

Arguments

`.x`	An object to test.

Value

TRUE if .x is of class "chronicle", FALSE if not.

make_log_df *Creates the log_df element of a chronicle object.*

Description

Creates the log_df element of a chronicle object.

Usage

```
make_log_df(
  ops_number = 1,
  success,
  fstring,
  args,
  res_pure,
  start = Sys.time(),
  end = Sys.time(),
  .g = (function(x) NA),
  diff_obj = NULL
)
```

Arguments

ops_number	Tracks the number of the operation in a chain of operations.
success	Did the operation succeed?
fstring	The function call.
args	The arguments of the call.
res_pure	The result of the purely call.
start	Starting time.
end	Ending time.
.g	Optional. A function to apply to the intermediary results for monitoring purposes. Defaults to returning NA.
diff_obj	Optional. Output of the diff parameter in record().

Value

A tibble containing the log.

pick *Retrieve an element from a chronicle object.*

Description

Retrieve an element from a chronicle object.

Usage

```
pick(.c, .e)
```

Arguments

.c	A chronicle object.
.e	Element of interest to retrieve, one of "value" or "log_df".

Value

The `value` or `log_df` element of the chronicle object .c.

Examples

```
r_sqrt <- record(sqrt)
r_exp <- record(exp)
3 |> r_sqrt() %>=% r_exp() |> pick("value")
```

print.chronicle *Print method for chronicle objects.*

Description

Print method for chronicle objects.

Usage

```
## S3 method for class 'chronicle'
print(x, ...)
```

Arguments

x	A chronicle object.
...	Unused.

Details

`chronicle` object are, at their core, lists with the following elements:

- "$value": an object of type `maybe` containing the result of the computation (see the "Maybe monad" vignette for more details on `maybe`s).
- "$log_df": a `data.frame` object containing the printed object's log information. This object is used by `read_log()` to generate a human-readable log.

`print.chronicle()` prints the object on screen and shows:

- the "$value" using its `print()` method (for example, if "$value" is a `data.frame` object, `print.data.frame()` will be used).
- a message indicating to the user how to get "$value" out of the `chronicle` object and how to read the object's log.

Value

No return value, called for side effects (printing the object on screen).

---

purely *Capture all errors, warnings and messages.*

Description

Capture all errors, warnings and messages.

Usage

```
purely(.f, strict = 2)
```

Arguments

`.f`	A function to decorate.
`strict`	Controls if the decorated function should catch only errors (1), errors and warnings (2, the default) or errors, warnings and messages (3).

Value

A function which returns a list. The first element of the list, `$value`, is the result of the original function `.f` applied to its inputs. The second element, `$log` is `NULL` in case everything goes well. In case of error/warning/message, `$value` is NA and `$log` holds the message. `purely()` is used by `record()` to allow the latter to handle errors.

Examples

```
purely(log)(10)
purely(log)(-10)
purely(log, strict = 1)(-10) # This produces a warning, so with strict = 1 nothing gets captured.
```

read_log *Reads the log of a chronicle.*

Description

Reads the log of a chronicle.

Usage

```
read_log(.c)
```

Arguments

.c	A chronicle object.

Value

The log of the object.

Examples

```
## Not run:
read_log(chronicle_object)

## End(Not run)
```

record *Decorates a function to output objects of type* chronicle.

Description

Decorates a function to output objects of type chronicle.

Usage

```
record(.f, .g = (function(x) NA), strict = 2, diff = "none")
```

Arguments

.f	A function to decorate.
.g	Optional. A function to apply to the intermediary results for monitoring purposes. Defaults to returning NA.
strict	Controls if the decorated function should catch only errors (1), errors and warnings (2, the default) or errors, warnings and messages (3).
diff	Whether to show the diff between the input and the output ("full"), just a summary of the diff ("summary"), or none ("none", the default)

Details

To chain multiple decorated function, use `bind_record()` or `%>=%`. If the `diff` parameter is set to "full", `diffobj::diffObj()` (or `diffobj::summary(diffobj::diffObj()`, if diff is set to "summary") gets used to provide the diff between the input and the output. This diff can be found in the `log_df` element of the result, and can be viewed using `check_diff()`.

Value

A function which returns objects of type `chronicle`. `chronicle` objects carry several elements: a `value` which is the result of the function evaluated on its inputs and a second object called `log_df`. `log_df` contains logging information, which can be read using `read_log()`. `log_df` is a data frame with columns: outcome, function, arguments, message, start_time, end_time, run_time, g and diff_obj.

Examples

```
record(sqrt)(10)
```

`record_many` *Decorate a list of functions*

Description

Decorate a list of functions

Usage

```
record_many(list_funcs, .g = (function(x) NA), strict = 2, diff = "none")
```

Arguments

`list_funcs`	A list of function names, as strings.
`.g`	Optional. Defaults to a function which returns NA.
`strict`	Controls if the decorated function should catch only errors (1), errors and warnings (2, the default) or errors, warnings and messages (3).
`diff`	Whether to show the diff between the input and the output ("full"), just a summary of the diff ("summary"), or none ("none", the default)

Details

Functions must be entered as strings of the form "function" or "package::function". The code gets generated and copied into the clipboard. The code can then be pasted into the text editor. On GNU/Linux systems, you might get the following error message on first use: "Error in : Clipboard on X11 requires that the DISPLAY envvar be configured". This is an error message from `clipr::write_clip()`, used by `record_many()` to put the generated code into the system's clipboard. To solve this issue, run `echo $DISPLAY` in the system's shell. This command should return a string like ":0". Take note of this string. In your .Rprofile, put the following command: Sys.setenv(DISPLAY = ":0") and restart the R session. `record_many()` should now work.

Value

Puts a string into the systems clipboard.

Examples

```
## Not run:
list_funcs <- list("exp", "dplyr::select", "exp")
record_many(list_funcs)

## End(Not run)
```

`%>=%` *Pipe a chronicle object to a decorated function.*

Description

Pipe a chronicle object to a decorated function.

Usage

```
.c %>=% .f
```

Arguments

`.c`	A value returned by record.
`.f`	A chronicle function to apply to the returning value of .c.

Value

A chronicle object.

Examples

```
r_sqrt <- record(sqrt)
r_exp <- record(exp)
3 |> r_sqrt() %>=% r_exp()
```

Index