Package 'bupaR'

May 8, 2026

Type Package

Title Business Process Analysis in R

Version 1.0.1

Description Comprehensive Business Process Analysis toolkit. Creates S3-class for event log objects, and related handler functions. Imports related packages for filtering event data, computation of descriptive statistics, handling of 'Petri Net' objects and visualization of process maps. See also packages 'edeaR','processmapR', 'eventdataR' and 'processmonitR'.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 3.5.0)

Imports magrittr, dplyr, data.table, shiny, miniUI, pillar, purrr, tidyr, tibble, glue, forcats, rlang (>= 1.0.0), cli (>= 3.2.0), eventdataR (>= 0.2.0), stringr, stringi, lubridate, lifecycle, ggplot2

URL https://bupar.net/, https://github.com/bupaverse/bupaR/, https://bupaverse.github.io/bupaR/

Suggests covr, lintr, edeaR, testthat (>= 3.1.3)

Config/testthat/edition 3

NeedsCompilation no

Author Benoît Depaire [cre], Gert Janssenswillen [aut], Gerard van Hulzen [ctb], Felix Mannhardt [ctb], Niels Martin [ctb], Greg Van Houdt [ctb], Hasselt University [cph]

Maintainer Benoît Depaire <benoit.depaire@uhasselt.be>

Repository CRAN

Date/Publication 2026-01-27 06:10:41 UTC

Contents

activities *Activities*

Description

Returns a tibble containing a list of all activity types in the event log, with their absolute and relative frequency

Usage

```
activities(log, ...)

## S3 method for class 'activitylog'
activities(log, ...)

## S3 method for class 'grouped_log'
activities(log, ...)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
...	Unused.

Methods (by class)

- activities(activitylog): Compute activity frequencies
- activities(grouped_log): Compute activity frequencies

See Also

activity_id,activity_instance_id, eventlog

activitylog *Create activity log*

Description

Create activity log

Usage

```
activitylog(activitylog, case_id, activity_id, resource_id, timestamps, order)
```

Arguments

activitylog	The data object to be used as activity log. This can be a data.frame or tibble.
case_id	The case classifier of the activity log. A character vector containing variable names of length 1 or more.
activity_id	The activity classifier of the activity log. A character vector containing variable names of length 1 or more.
resource_id	The resource identifier of the activity log. A character vector containing variable names of length 1 or more.
timestamps	The columns with timestamps refering to different lifecycle events. A character vector of 1 or more. These should have one of the following names: "schedule","assign","reassign","start","suspend","resume","abort_activity","abort_case","complete","manualski These columns should be of the Date or POSIXct class.
order	Configure how to handle sort events with equal timestamps: auto will use the order in the original data, alphabetical will sort the activity labels by alphabet, sorted will assume that the data frame is already correctly sorted and has a column '.order', providing a column name will use this column for ordering (can be numeric of character). The latter will never overrule timestamp orderings.

---

activity_id *Activity classifier*

Description

Get the activity classifier of an object of class eventlog.

Usage

```
activity_id(x)

## Default S3 method:
activity_id(x)

## S3 method for class 'log_mapping'
activity_id(x)
```

Arguments

x	log: Object of class eventlog or activitylog, or mapping.

Methods (by class)

- activity_id(default): Retrieve activity identifier from log
- activity_id(log_mapping): Retrieve activity identifier from mapping

See Also

eventlog, activitylog, mapping

Other Classifiers: activity_instance_id(), case_id(), lifecycle_id(), resource_id(), set_activity_id(), set_activity_instance_id(), set_case_id(), set_lifecycle_id(), set_resource_id(), set_timestamp(), timestamp(), timestamps()

---

activity_instance_id *Activity instance classifier*

Description

Get the activity instance classifier of an object of class eventlog.

Usage

```
activity_instance_id(x)

## S3 method for class 'eventlog'
activity_instance_id(x)

## S3 method for class 'eventlog_mapping'
activity_instance_id(x)

## S3 method for class 'activitylog'
activity_instance_id(x)

## S3 method for class 'activitylog_mapping'
activity_instance_id(x)
```

Arguments

x	An eventlog or eventlog_mapping

Methods (by class)

- activity_instance_id(eventlog): Retrieve activity instance identifier from eventlog
- activity_instance_id(eventlog_mapping): Retrieve activity instance identifier from eventlog mapping
- activity_instance_id(activitylog): Retrieve activity instance identifier from activitylog
- activity_instance_id(activitylog_mapping): Retrieve activity instance identifier from activitylog mapping

See Also

Other Classifiers: activity_id(), case_id(), lifecycle_id(), resource_id(), set_activity_id(), set_activity_instance_id(), set_case_id(), set_lifecycle_id(), set_resource_id(), set_timestamp(), timestamp(), timestamps()

`activity_labels` *Get vector of activity labels*

Description

Retrieve a vector containing all unique activity labels

Usage

```
activity_labels(log)

## S3 method for class 'log'
activity_labels(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- activity_labels(log): Retrieve activity labels

`act_collapse` *Collapse activity labels of a sub process into a single activity*

Description

Collapse activity labels of a sub process into a single activity

Usage

```
act_collapse(log, ..., method)

## S3 method for class 'eventlog'
act_collapse(log, ..., method = c("entry_points", "consecutive"))

## S3 method for class 'activitylog'
act_collapse(log, ..., method = c("entry_points", "consecutive"))

## S3 method for class 'grouped_log'
act_collapse(log, ..., method = c("entry_points", "consecutive"))
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
...	A series of named character vectors. The activity labels in each vector will be collapsed into one activity with the name of the vector.
method	Defines how activities are collapsed: "entry_points" heuristically learns which of the specified activities occur at the start and end of the subprocess and collapses accordingly. "consecutive" collapses consecutive sequences of the activities.

Details

There are different strategies to collapse activity labels (argument ´method´). The "entry_points" method aims to learn the start and end activities of the sub process, by looking at the first and last activity in each case over the whole log. Subsequently, it will create a new instance of the sub process each time there is an end activity followed by a start activity. This strategy will not take into account other activities happening in the mean time. The "consecutive" method will create an instance each time a new sequence of sub activities is started. This strategy will thus only take into account interruptions of the other activity labels.

Methods (by class)

- act_collapse(eventlog): Collapse activity labels of a subprocess into a single activity
- act_collapse(activitylog): Collapse activity labels of a subprocess into a single activity
- act_collapse(grouped_log): Collapse activity labels of a subprocess into a single activity

See Also

Other Activity processing functions: act_recode(), act_unite()

---

act_recode *Recode activity labels*

Description

Recode one or more activity labels through specifying their old and new label

Usage

```
act_recode(log, ...)

## S3 method for class 'log'
act_recode(log, ...)

## S3 method for class 'grouped_log'
act_recode(log, ...)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
...	A sequence of named character vectors of length one where the names gives the new label and the value gives the old label. Labels not mentioned will be left unchanged.

Methods (by class)

- act_recode(log): Recode activity labels of event log
- act_recode(grouped_log): Recode activity labels of event log

See Also

eventlog, activity_id, act_unite

Other Activity processing functions: act_collapse(), act_unite()

---

act_unite *Unite activity labels*

Description

Recode two or different more activity labels two a uniform activity label

Usage

```
act_unite(log, ...)

## S3 method for class 'log'
act_unite(log, ...)

## S3 method for class 'grouped_log'
act_unite(log, ...)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
...	A series of named character vectors. The activity labels in each vector will be replaced with the name.

Methods (by class)

- act_unite(log): Unite activity labels in event log
- act_unite(grouped_log): Unite activity labels of event log

See Also

eventlog, activity_id, act_recode

Other Activity processing functions: act_collapse(), act_recode()

---

add_end_activity *Add Artificial Start/End Activities*

Description

Adds an artificial start or end activity to each case with the specified label.

Usage

```
add_end_activity(log, label = "End")

add_start_activity(log, label = "Start")

## S3 method for class 'eventlog'
add_end_activity(log, label = "End")

## S3 method for class 'activitylog'
add_end_activity(log, label = "End")

## S3 method for class 'grouped_log'
add_end_activity(log, label = "End")

## S3 method for class 'eventlog'
add_start_activity(log, label = "Start")

## S3 method for class 'activitylog'
add_start_activity(log, label = "Start")

## S3 method for class 'grouped_log'
add_start_activity(log, label = "Start")
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
label	character: Start (default "Start") or end (default "End") activity label. This must be an activity label that is not already present in log.

Methods (by class)

- add_end_activity(eventlog): Adds end activity to an eventlog.
- add_end_activity(activitylog): Adds end activity to an activitylog.
- add_end_activity(grouped_log): Adds end activity to a grouped_log.

Functions

- add_start_activity(eventlog): Adds start activity to an eventlog.
- add_start_activity(activitylog): Adds start activity to an activitylog.
- add_start_activity(grouped_log): Adds start activity to a grouped_log.

---

arrange *Arrange log*

Description

Arrange log

Usage

```
arrange(.data, ..., .by_group = FALSE)

## S3 method for class 'grouped_eventlog'
arrange(.data, ...)

## S3 method for class 'activitylog'
arrange(.data, ...)
```

Arguments

.data	log: Object of class eventlog or activitylog.
...	Additional arguments passed to dplyr
.by_group	If TRUE, will sort first by grouping variable. Applies to grouped data frames only.

Methods (by class)

- arrange(grouped_eventlog): Arrange an eventlog by group, maintaining all groups
- arrange(activitylog): Arrange an activitylog

as.grouped.data.frame *as.grouped.data.frame*

Description

as.grouped.data.frame

Usage

```
as.grouped.data.frame(data, groups)
```

Arguments

data	Data
groups	Names of grouping variables as character vector (e.g. by using dplyr::group_vars

assign_instance_id *Assign activity instance identifier to events*

Description

Apply heuristics to create an activity instance identifier, so that an eventlog can be made.

Usage

```
assign_instance_id(eventlog, case_id, activity_id, timestamp, lifecycle_id)
```

Arguments

eventlog	data.frame with events
case_id	Case identifier
activity_id	Activity identifier
timestamp	Timestamp
lifecycle_id	Lifecycle identifier

See Also

Other Eventlog construction helpers: convert_timestamps()

bupaR *bupaR - Business Process Analysis in R*

Description

Functionalities for process analysis in R. This packages implements an S3-class for event log objects, and related handler functions. Imports related packages for subsetting event data, computation of descriptive statistics, handling of Petri Net objects and visualization of process maps.

cases *Cases*

Description

Provides a fine-grained summary of an event log with characteristics for each case: the number of events, the number of activity types, the timespan, the trace, the duration, and the first and last event type.

Usage

```
cases(log, ...)

## S3 method for class 'log'
cases(log, ...)

## S3 method for class 'eventlog'
cases(log, ...)

## S3 method for class 'activitylog'
cases(log, ...)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
...	Other (optional) arguments passed on to methods. See durations for more options.

Methods (by class)

- cases(log): Construct list of cases in a log.
- cases(eventlog): Construct list of cases in an eventlog.
- cases(activitylog): Construct list of cases in a activitylog.

See Also

case_list,durations

case_id *Case classifier*

Description

Get the case classifier of an object of class `eventlog`

Usage

```
case_id(x)

## S3 method for class 'eventlog'
case_id(x)

## S3 method for class 'eventlog_mapping'
case_id(x)

## S3 method for class 'activitylog'
case_id(x)

## S3 method for class 'activitylog_mapping'
case_id(x)
```

Arguments

x	log: Object of class eventlog or activitylog, or mapping.

Methods (by class)

- `case_id(eventlog)`: Retrieve case identifier from eventlog
- `case_id(eventlog_mapping)`: Retrieve case identifier from eventlog mapping
- `case_id(activitylog)`: Retrieve case identifier from activitylog
- `case_id(activitylog_mapping)`: Retrieve case identifier from activitylog mapping

See Also

eventlog, activitylog, mapping

Other Classifiers: activity_id(), activity_instance_id(), lifecycle_id(), resource_id(), set_activity_id(), set_activity_instance_id(), set_case_id(), set_lifecycle_id(), set_resource_id(), set_timestamp(), timestamp(), timestamps()

case_labels *Case labels*

Description

Retrieve a vector containing all unique case labels

Usage

```
case_labels(log)

## S3 method for class 'log'
case_labels(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- `case_labels(log)`: Retrieve case labels from log

case_list *Case list*

Description

Construct list of cases

Usage

```
case_list(log, .keep_trace_list)

## S3 method for class 'eventlog'
case_list(log, .keep_trace_list = FALSE)

## S3 method for class 'activitylog'
case_list(log, .keep_trace_list = FALSE)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
.keep_trace_list	Logical (default is FALSE): If TRUE, keeps the trace as a list. If FALSE, only the concatenated string representation of the trace is kept.

Methods (by class)

- `case_list(eventlog)`: Return case list
- `case_list(activitylog)`: Return case list

---

convert_timestamps *Convert timestamp format*

Description

Function converting the timestamps in the data frame to the appropriate format.

Usage

```
convert_timestamps(x, columns, format)
```

Arguments

`x`	Data.frame containing events or activities.
`columns`	A character vector with one or more names of columns to convert
`format`	The format of the timestamps in the original dataset (either ymd_hms, dmy_hms, ymd_hm, ymd, dmy, dmy, ...). To be provided without quotation marks!

Value

Data.frame with converted timestamps

See Also

Other Eventlog construction helpers: `assign_instance_id()`

---

count *Count log*

Description

Count log

Usage

```
count(x, ..., wt = NULL, sort = FALSE, name = NULL)

## S3 method for class 'log'
count(x, ...)

## S3 method for class 'grouped_log'
count(x, ...)
```

Arguments

x	log: Object of class eventlog or activitylog.
...	Additional arguments passed to dplyr
wt	<data-masking> Frequency weights. Can be NULL or a variable: • If NULL (the default), counts the number of rows in each group. • If a variable, computes sum(wt) for each group.
sort	If TRUE, will show the largest groups at the top.
name	The name of the new column in the output. If omitted, it will default to n. If there's already a column called n, it will use nn. If there's a column called n and nn, it'll use nnn, and so on, adding ns until it gets a new name.

Methods (by class)

- count(log): Count log
- count(grouped_log): Count grouped log

---

detect_resource_inconsistencies

Detect resource inconsistencies

Description

Function to detect inconsistencies in resource information between related events.

Usage

```
detect_resource_inconsistencies(eventlog, filter_condition)
```

Arguments

eventlog	Event log object
filter_condition	Condition that is used to extract a subset of the activity log prior to the application of the function

durations *Durations*

Description

Computes the throughput times of each case. Throughput time is defined as the interval between the start of the first event and the completion of the last event.

Usage

```
durations(log, units = c("auto", "secs", "mins", "hours", "days", "weeks"))
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
units	character (default "auto"): The time unit in which the throughput times should be reported. Should be one of the following values: "auto" (default), "secs", "mins", "hours", "days", "weeks". See also the units argument of difftime.

eventlog *Eventlog*

Description

A function to instantiate an object of class eventlog by specifying a data.frame or tibble and appropriate case, activity and timestamp classifiers.

Usage

```
eventlog(
  eventlog,
  case_id,
  activity_id,
  activity_instance_id,
  lifecycle_id,
  timestamp,
  resource_id,
  order,
  validate
)

ieventlog(eventlog)
```

Arguments

eventlog	The data object to be used as event log. This can be a data.frame or tibble.
case_id	The case classifier of the event log. A character vector containing variable names of length 1 or more.
activity_id	The activity classifier of the event log. A character vector containing variable names of length 1 or more.
activity_instance_id	The activity instance classifier of the event log.
lifecycle_id	The life cycle classifier of the event log.
timestamp	The timestamp of the event log. Should refer to a Date or POSIXct field.
resource_id	The resource identifier of the event log. A character vector containing variable names of length 1 or more.
order	Configure how to handle sort events with equal timestamps: auto will use the order in the original data, alphabetical will sort the activity labels by alphabet, sorted will assume that the data frame is already correctly sorted and has a column '.order', providing a column name will use this column for ordering (can be numeric of character). The latter will never overrule timestamp orderings.
validate	When TRUE some basic checks are run on the contents of the event log such as that activity instances are not connected to more than one case or activity. Using FALSE improves the performance by skipping those checks.

See Also

case_id, activity_id, activity_instance_id,lifecycle_id, timestamp

Examples

```
## Not run:
data <- data.frame(case = rep("A",5),
activity_id = c("A","B","C","D","E"),
activity_instance_id = 1:5,
lifecycle_id = rep("complete",5),
timestamp = 1:5,
resource = rep("resource 1", 5))
eventlog(data,case_id = "case",
activity_id = "activity_id",
activity_instance_id = "activity_instance_id",
lifecycle_id = "lifecycle_id",
timestamp = "timestamp",
resource_id = "resource")

## End(Not run)
```

`events_to_activitylog` *Events to activities*

Description

[Deprecated] Create an activity log starting from an event log or regular data.frame. This function is deprecated and replaced by the function `activitylog` (for dataframe) and `to_activitylog` for eventlogs.

Usage

```
events_to_activitylog(
  eventlog,
  case_id,
  activity_id,
  activity_instance_id,
  lifecycle_id,
  timestamp,
  resource_id,
  ...
)
```

Arguments

`eventlog`	The event log to be converted. An object of class `eventlog` or `data.frame`
`case_id`	If eventlog is data.frame, the case classifier of the event log. A character vector containing variable names of length 1 or more.
`activity_id`	If eventlog is data.frame, the activity classifier of the event log. A character vector containing variable names of length 1 or more.
`activity_instance_id`	If eventlog is data.frame, the activity instance classifier of the event log.
`lifecycle_id`	If eventlog is data.frame, the life cycle classifier of the event log.
`timestamp`	If eventlog is data.frame, the timestamp of the event log. Should refer to a Date or POSIXct field.
`resource_id`	If eventlog is data.frame, the resource identifier of the event log. A character vector containing variable names of length 1 or more.
`...`	Additional argments, i.e. for fixing resource inconsistencies

filter *Filter event log*

Description

Filter event log

Usage

```
filter(.data, ..., .by = NULL, .preserve = FALSE)
```

Arguments

.data	log: Object of class eventlog or activitylog.
...	Additional arguments passed to dplyr
.by	**[Experimental]** <tidy-select> Optionally, a selection of columns to group by for just this operation, functioning as an alternative to group_by(). For details and examples, see ?dplyr_by.
.preserve	Relevant when the .data input is grouped. If .preserve = FALSE (the default), the grouping structure is recalculated based on the resulting data, otherwise the grouping is kept as is.

first_n *first_n*

Description

Select first n activity instances.

Usage

```
first_n(log, n)

## S3 method for class 'eventlog'
first_n(log, n)

## S3 method for class 'activitylog'
first_n(log, n)

## S3 method for class 'grouped_log'
first_n(log, n)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
n	integer: The number of activity instances to select.

Methods (by class)

- first_n(eventlog): Select first n activity instances of an eventlog.
- first_n(activitylog): Select first n activity instances of an activitylog.
- first_n(grouped_log): Select first n activity instances of a grouped_log.

---

fix_resource_inconsistencies

Fix resource inconsistencies

Description

Fix resource inconsistencies

Usage

```
fix_resource_inconsistencies(
  eventlog,
  filter_condition,
  overwrite_missings,
  detected_problems,
  details
)

## S3 method for class 'activitylog'
fix_resource_inconsistencies(
  eventlog,
  filter_condition = NULL,
  overwrite_missings = FALSE,
  detected_problems = NULL,
  details = TRUE
)

## S3 method for class 'eventlog'
fix_resource_inconsistencies(
  eventlog,
  filter_condition = NULL,
  overwrite_missings = FALSE,
  detected_problems = NULL,
  details = TRUE
)
```

Arguments

`eventlog`	Event log object
`filter_condition`	Condition that is used to extract a subset of the activity log prior to the application of the function
`overwrite_missings`	If events are missing, overwrite the resource if other events within activity instance are performed by single resource. Default FALSE.
`detected_problems`	If available, the problems detected that need to be fixed. If not available, the function detect_resource_inconsistenties will be called.
`details`	Show details

Methods (by class)

- `fix_resource_inconsistencies(activitylog)`: activitylog Fix activitylog
- `fix_resource_inconsistencies(eventlog)`: eventlog Fix eventlog

`grouped_activitylog` *Grouped activitylog object*

Description

Lorem ipsum

`grouped_eventlog` *Grouped eventlog object*

Description

Lorem ipsum

`grouped_log` *Grouped log object*

Description

Lorem ipsum

group_by *Group event log*

Description

Group event log

Usage

```
group_by(.data, ..., .add = FALSE, .drop = group_by_drop_default(.data))
```

Arguments

.data	log: Object of class eventlog or activitylog.
...	Variables to group on
.add	Add grouping variables to existing ones
.drop	Drop groups formed by factor levels that don't appear in the data? The default is TRUE except when .data has been previously grouped with .drop = FALSE. See group_by_drop_default() for details.

group_by_activity *Group event log on activity id*

Description

Group an event log by activity identifier

Usage

```
group_by_activity(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

group_by_activity_instance *Group event log on activity instance id*

Description

Group an event log by activity instance identifier

Usage

```
group_by_activity_instance(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

group_by_case *Group event log on case id*

Description

Group an event log by case identifier

Usage

```
group_by_case(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

group_by_ids *Group log on identifiers*

Description

Group log on identifiers

Usage

```
group_by_ids(log, ...)

## S3 method for class 'log'
group_by_ids(log, ...)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
...	One or more of the following: activity_id, case_id, activity_instance_id, resource_id, lifecycle_id

Value

Grouped log

Methods (by class)

- group_by_ids(log): Group log on identifiers

group_by_resource *Group event log on resource id*

Description

Group an event log by resource identifier

Usage

```
group_by_resource(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

group_by_resource_activity *Group event log on resource and activity id*

Description

Group an event log by resource and activity identifier

Usage

```
group_by_resource_activity(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

group_by_trace *Group a log on trace*

Description

Group a log by trace

Usage

```
group_by_trace(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

is.log *Test if the Object is a Log*

Description

This function returns TRUE if x inherits from the specified class, and FALSE for all other objects.

Usage

```
is.log(x)

is.eventlog(x)

is.activitylog(x)

is.grouped_log(x)

is.grouped_eventlog(x)

is.grouped_activitylog(x)
```

Arguments

x	Any R object.

Value

is.log returns TRUE if the object inherits from the log class, otherwise FALSE.

is.eventlog returns TRUE if the object inherits from the eventlog class, otherwise FALSE.

is.actvitylog returns TRUE if the object inherits from the activitylog class, otherwise FALSE.

is.grouped_log returns TRUE if the object inherits from the grouped_log class, otherwise FALSE.

is.grouped_eventlog returns TRUE if the object inherits from the grouped_eventlog class, otherwise FALSE.

is.grouped_activitylog returns TRUE if the object inherits from the grouped_activitylog class, otherwise FALSE.

See Also

log,eventlog,activitylog,grouped_log,grouped_eventlog,grouped_activitylog

last_n *last_n*

Description

Select last n activity instances

Usage

```
last_n(log, n)

## S3 method for class 'eventlog'
last_n(log, n)

## S3 method for class 'activitylog'
last_n(log, n)

## S3 method for class 'grouped_log'
last_n(log, n)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
n	integer: The number of activity instances to select.

Methods (by class)

- last_n(eventlog): Select last n activity instances of an eventlog.
- last_n(activitylog): Select last n activity instances of an activitylog.
- last_n(grouped_log): Select last n activity instances of a grouped_log.

lifecycles *Life cycles*

Description

Returns a tibble containing a list of all life cycle types in the log, with their absolute and relative frequency (# events).

Usage

```
lifecycles(log)

## S3 method for class 'eventlog'
lifecycles(log)

## S3 method for class 'grouped_eventlog'
lifecycles(log)

## S3 method for class 'activitylog'
lifecycles(log)

## S3 method for class 'grouped_activitylog'
lifecycles(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- lifecycles(eventlog): Generate lifecycle list for an eventlog.
- lifecycles(grouped_eventlog): Generate lifecycle list for a grouped_eventlog.
- lifecycles(activitylog): Generate lifecycle list for an activitylog.
- lifecycles(grouped_activitylog): Generate lifecycle list for an grouped_activitylog.

See Also

lifecycle_id

---

lifecycle_id *Life cycle classifier*

Description

Get the life_cycle_id of an object of class eventlog

Usage

```
lifecycle_id(x)

## S3 method for class 'eventlog'
lifecycle_id(x)

## S3 method for class 'eventlog_mapping'
lifecycle_id(x)
```

Arguments

x	eventlog: Object of class eventlog, or mapping.

Methods (by class)

- lifecycle_id(eventlog): Retrieve lifecycle identifier from eventlog
- lifecycle_id(eventlog_mapping): Retrieve lifecycle identifier from eventlog mapping

See Also

Other Classifiers: activity_id(), activity_instance_id(), case_id(), resource_id(), set_activity_id(), set_activity_instance_id(), set_case_id(), set_lifecycle_id(), set_resource_id(), set_timestamp(), timestamp(), timestamps()

---

lifecycle_labels *Get vector of lifecycle labels.*

Description

Retrieve a vector containing all unique lifecycle labels.

Usage

```
lifecycle_labels(log)

## S3 method for class 'eventlog'
lifecycle_labels(log)

## S3 method for class 'activitylog'
lifecycle_labels(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- lifecycle_labels(eventlog): Retrieve lifecycle labels from an eventlog.
- lifecycle_labels(activitylog): Retrieve lifecycle labels from an activitylog.

See Also

lifecycle_id

log *Log object*

Description

Lorem ipsum

mapping *Mapping*

Description

Prints the mapping of an event log object.

Usage

```
mapping(log)

## S3 method for class 'eventlog'
mapping(log)

## S3 method for class 'activitylog'
mapping(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- mapping(eventlog): Retrieve identifier mapping from eventlog
- mapping(activitylog): Retrieve identifier mapping from activitylog

mutate *Mutate event log*

Description

Mutate event log

Usage

```
mutate(.data, ...)
```

Arguments

.data	A data frame, data frame extension (e.g. a tibble), or a lazy data frame (e.g. from dbplyr or dtplyr). See *Methods*, below, for more details.
...	Additional arguments passed to dplyr

n_activities *n_activities*

Description

Returns the number of activities in an event log

Usage

```
n_activities(log)

## S3 method for class 'log'
n_activities(log)

## S3 method for class 'grouped_log'
n_activities(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- n_activities(log): Count the number of activities in a log
- n_activities(grouped_log): Count the number of activities for a grouped log

See Also

Other Counters: n_activity_instances(), n_cases(), n_events(), n_resources(), n_traces()

n_activity_instances *n_activity_instances*

Description

Returns the number of activity instances in an event log

Usage

```
n_activity_instances(log)

## S3 method for class 'eventlog'
n_activity_instances(log)

## S3 method for class 'grouped_eventlog'
n_activity_instances(log)

## S3 method for class 'activitylog'
n_activity_instances(log)

## S3 method for class 'grouped_activitylog'
n_activity_instances(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- n_activity_instances(eventlog): eventlog
- n_activity_instances(grouped_eventlog): grouped_eventlog
- n_activity_instances(activitylog): activitylog
- n_activity_instances(grouped_activitylog): grouped_activitylog

See Also

Other Counters: n_activities(), n_cases(), n_events(), n_resources(), n_traces()

n_cases *n_cases*

Description

Returns the number of cases in an event log.

Usage

```
n_cases(log)

## S3 method for class 'log'
n_cases(log)

## S3 method for class 'grouped_log'
n_cases(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- n_cases(log): Count number of cases in a log.
- n_cases(grouped_log): Count number of cases in a grouped_log.

See Also

Other Counters: n_activities(), n_activity_instances(), n_events(), n_resources(), n_traces()

n_events *n_events*

Description

Returns the number of events in an event log.

Usage

```
n_events(log)

## S3 method for class 'eventlog'
n_events(log)

## S3 method for class 'grouped_eventlog'
n_events(log)

## S3 method for class 'activitylog'
n_events(log)

## S3 method for class 'grouped_activitylog'
n_events(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- n_events(eventlog): Count number of events in an eventlog.
- n_events(grouped_eventlog): Count number of events in a grouped_eventlog.
- n_events(activitylog): Count number of events in an activitylog.
- n_events(grouped_activitylog): Count number of events in an grouped_activitylog.

See Also

Other Counters: n_activities(), n_activity_instances(), n_cases(), n_resources(), n_traces()

---

n_resources *n_resources*

Description

Returns the number of resources in an event log

Usage

```
n_resources(log)

## S3 method for class 'log'
n_resources(log)

## S3 method for class 'grouped_log'
n_resources(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- n_resources(log): Count number of resources in log
- n_resources(grouped_log): Count number of resources in grouped log

See Also

Other Counters: n_activities(), n_activity_instances(), n_cases(), n_events(), n_traces()

---

n_traces *n_traces*

Description

Returns the number of traces in an event log

Usage

```
n_traces(log)

## S3 method for class 'log'
n_traces(log)

## S3 method for class 'grouped_log'
n_traces(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- n_traces(log): Count number of traces for eventlog
- n_traces(grouped_log): Count number of traces for grouped eventlog

See Also

Other Counters: n_activities(), n_activity_instances(), n_cases(), n_events(), n_resources()

print.eventlog_mapping *Generic print function for mapping.*

Description

Generic print function for mapping.

Usage

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

Arguments

x	Mapping of eventlog or activitylog
...	Additional Arguments

print.log *Generic print function for eventlog*

Description

Generic print function for eventlog

Usage

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

Arguments

x	log: Object of class eventlog or activitylog.
...	Additional Arguments

rename *Rename log*

Description

Rename log

Usage

```
rename(.data, ...)

## S3 method for class 'log'
rename(.data, ...)

## S3 method for class 'grouped_log'
rename(.data, ...)
```

Arguments

.data	log: Object of class eventlog or activitylog.
...	Variables to rename. Use "new_name" = "old_name" to rename selected variables.

Methods (by class)

- rename(log): Rename log
- rename(grouped_log): Rename grouped log

resources *Resources*

Description

Returns a tibble containing a list of all resources in the event log, with their absolute and relative frequency

Usage

```
resources(log)

## S3 method for class 'eventlog'
resources(log)

## S3 method for class 'activitylog'
resources(log)

## S3 method for class 'grouped_log'
resources(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- resources(eventlog): Generate resource list for eventlog
- resources(activitylog): Generate resource list for activitylog
- resources(grouped_log): Compute activity frequencies

See Also

resource_id, eventlog

---

resource_id *Resource classifier*

Description

Get the resource classifier of an object of class eventlog.

Usage

```
resource_id(x)

## S3 method for class 'eventlog'
resource_id(x)

## S3 method for class 'eventlog_mapping'
resource_id(x)

## S3 method for class 'activitylog'
resource_id(x)

## S3 method for class 'activitylog_mapping'
resource_id(x)
```

Arguments

x	log: Object of class eventlog or activitylog, or mapping.

Methods (by class)

- resource_id(eventlog): Retrieve resource identifier from eventlog
- resource_id(eventlog_mapping): Retrieve resource identifier from eventlog mapping
- resource_id(activitylog): Retrieve resource identifier from activitylog
- resource_id(activitylog_mapping): Retrieve resource identifier from activitylog mapping

See Also

eventlog, mapping

Other Classifiers: activity_id(), activity_instance_id(), case_id(), lifecycle_id(), set_activity_id(), set_activity_instance_id(), set_case_id(), set_lifecycle_id(), set_resource_id(), set_timestamp(), timestamp(), timestamps()

---

resource_labels *Get vector of resource labels*

Description

Retrieve a vector containing all unique resource labels

Usage

```
resource_labels(log)

## Default S3 method:
resource_labels(log)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).

Methods (by class)

- resource_labels(default): Retrieve resource labels from eventlog

---

re_map *Re map*

Description

Construct an eventlog using an existing mapping.

Usage

```
re_map(x, mapping)
```

Arguments

x	log: Object of class eventlog or activitylog.
mapping	An existing mapping created by the mapping function

sample_n *Sample function for eventlog*

Description

Sample function for eventlog

Usage

```
sample_n(tbl, size, replace = FALSE, weight = NULL, .env = NULL, ...)

## S3 method for class 'log'
sample_n(tbl, size, replace = FALSE, weight = NULL, .env = NULL, ...)

## S3 method for class 'grouped_log'
sample_n(tbl, size, replace = FALSE, weight = NULL, .env = NULL, ...)
```

Arguments

tbl	Event log
size	integer: Number of cases to sample
replace	logical (default FALSE): Sample with replacement TRUE or without FALSE.
weight	Sampling weights. This must evaluate to a vector of non-negative numbers the same length as the input. Weights are automatically standardised to sum to 1.
.env	Deprecated; please don't use.
...	ignored

Methods (by class)

- sample_n(log): Sample n cases of eventlog
- sample_n(grouped_log): Stratified sampling of a grouped eventlog: sample n cases within each group

See Also

slice_sample

scale_fill_discrete_bupaR *bupaR color scales*

Description

bupaR color scales

Usage

```
scale_fill_discrete_bupaR(
  guide = "legend",
  na.value = "grey50",
  name = waiver()
)

scale_color_discrete_bupaR(
  guide = "legend",
  na.value = "grey50",
  name = waiver()
)

scale_fill_continuous_bupaR(
  guide = "colourbar",
  na.value = "grey50",
  name = waiver(),
  palette = c("green", "orange")
)

scale_color_continuous_bupaR(
  guide = "colourbar",
  na.value = "grey50",
  name = waiver()
)

scale_fill_gradient_bupaR(
  guide = "colourbar",
  na.value = "grey50",
  name = waiver()
)

scale_color_gradient_bupaR(
  guide = "colourbar",
  na.value = "grey50",
  name = waiver()
)
```

```
scale_fill_gradient2_bupaR(
  guide = "colourbar",
  na.value = "grey50",
  name = waiver(),
  midpoint = 0
)

scale_color_gradient2_bupaR(
  guide = "colourbar",
  na.value = "grey50",
  name = waiver(),
  midpoint = 0
)
```

Arguments

guide	Type of legend. Use "colourbar" for continuous colour bar, or "legend" for discrete colour legend.
na.value	Colour to use for missing values
name	The name of the scale. Used as the axis or legend title. If waiver(), the default, the name of the scale is taken from the first mapping used for that aesthetic. If NULL, the legend title will be omitted.
palette	Color palette to be used for scale_._continuous_bupaR. Can be "green" (default) or "orange".
midpoint	The midpoint (in data value) of the diverging scale. Defaults to 0.

---

select *Select event log*

Description

Select event log

Usage

```
## S3 method for class 'eventlog'
select(.data, ..., force_df = FALSE)
```

Arguments

.data	A data frame, data frame extension (e.g. a tibble), or a lazy data frame (e.g. from dbplyr or dtplyr). See *Methods*, below, for more details.
...	<tidy-select> One or more unquoted expressions separated by commas. Variable names can be used as if they were positions in the data frame, so expressions like x:y can be used to select a range of variables.
force_df	If TRUE, result will no longer be a event log when not all id columns are selected.

select_ids *Select identifiers from log*

Description

Select identifiers from log

Usage

```
select_ids(log, ...)

## S3 method for class 'log'
select_ids(log, ...)
```

Arguments

log	log: Object of class log, eventlog, or activitylog.
...	One or more of the following: activity_id, case_id, activity_instance_id, resource_id, lifecycle_id

Methods (by class)

- select_ids(log): Select identifiers from log

Examples

```
library(eventdataR)

patients %>% select_ids(activity_id, case_id)
```

set_activity_id *Set activity id of log*

Description

Set activity id of log

Usage

```
set_activity_id(log, activity_id)

## Default S3 method:
set_activity_id(log, activity_id)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
activity_id	New activity id

Methods (by class)

- set_activity_id(default): Set activity id

See Also

Other Classifiers: activity_id(), activity_instance_id(), case_id(), lifecycle_id(), resource_id(), set_activity_instance_id(), set_case_id(), set_lifecycle_id(), set_resource_id(), set_timestamp(), timestamp(), timestamps()

---

set_activity_instance_id

Set activity instance id of log

Description

Set activity instance id of log

Usage

```
set_activity_instance_id(log, activity_instance_id)

## S3 method for class 'eventlog'
set_activity_instance_id(log, activity_instance_id)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
activity_instance_id	New activity_instance id

Methods (by class)

- set_activity_instance_id(eventlog): Set activity_instance_id of eventlog

See Also

Other Classifiers: activity_id(), activity_instance_id(), case_id(), lifecycle_id(), resource_id(), set_activity_id(), set_case_id(), set_lifecycle_id(), set_resource_id(), set_timestamp(), timestamp(), timestamps()

set_case_id *Set case id of log*

Description

Set case id of log

Usage

```
set_case_id(log, case_id)

## Default S3 method:
set_case_id(log, case_id)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
case_id	New case id

Methods (by class)

- set_case_id(default): Set case id

See Also

Other Classifiers: activity_id(), activity_instance_id(), case_id(), lifecycle_id(), resource_id(), set_activity_id(), set_activity_instance_id(), set_lifecycle_id(), set_resource_id(), set_timestamp(), timestamp(), timestamps()

set_lifecycle_id *Set lifecycle id of log*

Description

Set lifecycle id of log

Usage

```
set_lifecycle_id(log, lifecycle_id)

## Default S3 method:
set_lifecycle_id(log, lifecycle_id)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
lifecycle_id	New lifecycle id. Can be multiple in case of activitylog

Methods (by class)

- set_lifecycle_id(default): Set lifecycle id

See Also

Other Classifiers: activity_id(), activity_instance_id(), case_id(), lifecycle_id(), resource_id(), set_activity_id(), set_activity_instance_id(), set_case_id(), set_resource_id(), set_timestamp(), timestamp(), timestamps()

---

set_resource_id *Set resource id of log*

Description

Set resource id of log

Usage

```
set_resource_id(log, resource_id)

## Default S3 method:
set_resource_id(log, resource_id)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
resource_id	New resource id

Methods (by class)

- set_resource_id(default): Set resource id

See Also

Other Classifiers: activity_id(), activity_instance_id(), case_id(), lifecycle_id(), resource_id(), set_activity_id(), set_activity_instance_id(), set_case_id(), set_lifecycle_id(), set_timestamp(), timestamp(), timestamps()

set_timestamp *Set timestamp of eventlog*

Description

Set timestamp of eventlog

Usage

```
set_timestamp(log, timestamp)

## S3 method for class 'eventlog'
set_timestamp(log, timestamp)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
timestamp	New timestamp

Methods (by class)

- set_timestamp(eventlog): Set timestamp of eventlog

See Also

Other Classifiers: activity_id(), activity_instance_id(), case_id(), lifecycle_id(), resource_id(), set_activity_id(), set_activity_instance_id(), set_case_id(), set_lifecycle_id(), set_resource_id(), timestamp(), timestamps()

simple_eventlog *Simple Eventlog*

Description

[Superseded]

A function to instantiate an object of class eventlog by specifying a data.frame or tibble and the minimally required case identifier, activity identifier and timestamp.

This function is superseded by the introduction of the activitylog format. Eventlogs in this 'simple' format can be seen as log of activities, and be created with activitylog(). If required, the resulting activity log can be transformed back to the eventlog format using to_eventlog.

Usage

```
simple_eventlog(
  eventlog,
  case_id = NULL,
  activity_id = NULL,
  timestamp = NULL,
  resource_id = NULL,
  order = "auto",
  return_type = c("eventlog", "activitylog")
)

isimple_eventlog(eventlog)
```

Arguments

eventlog	The data object to be used as event log. This can be a data.frame or tibble.
case_id	The case classifier of the event log.
activity_id	The activity classifier of the event log.
timestamp	The timestamp of the event log.
resource_id	The resource classifier of the event log (optional).
order	Configure how to handle sort events with equal timestamps: auto will use the order in the original data, alphabetical will sort the activity labels by alphabet, sorted will assume that the data frame is already correctly sorted and has a column '.order', providing a column name will use this column for ordering (can be numeric of character). The latter will never overrule timestamp orderings.
return_type	Whether to return eventlog (default) or activitylog object.

See Also

eventlog,case_id, activity_id, activity_instance_id,lifecycle_id, timestamp

Examples

```
## Not run:
data <- data.frame(case = rep("A",5),
activity_id = c("A","B","C","D","E"),
timestamp = date_decimal(1:5))
simple_eventlog(data,case_id = "case",
activity_id = "activity_id",
timestamp = "timestamp")

## End(Not run)
```

slice *Slice function for event log*

Description

Slice function for event log

Usage

```
slice(.data, ..., .by = NULL, .preserve = FALSE)

## S3 method for class 'log'
slice(.data, ...)

## S3 method for class 'grouped_log'
slice(.data, ...)

## S3 method for class 'eventlog'
slice_activities(.data, ...)

## S3 method for class 'activitylog'
slice_activities(.data, ...)

## S3 method for class 'grouped_log'
slice_activities(.data, ...)

## S3 method for class 'eventlog'
slice_events(.data, ...)

## S3 method for class 'grouped_eventlog'
slice_events(.data, ...)
```

Arguments

.data	A data frame, data frame extension (e.g. a tibble), or a lazy data frame (e.g. from dbplyr or dtplyr). See *Methods*, below, for more details.
...	Additional arguments passed to dplyr
.by	**[Experimental]** <tidy-select> Optionally, a selection of columns to group by for just this operation, functioning as an alternative to group_by(). For details and examples, see ?dplyr_by.
.preserve	Relevant when the .data input is grouped. If .preserve = FALSE (the default), the grouping structure is recalculated based on the resulting data, otherwise the grouping is kept as is.

Methods (by class)

- `slice(log)`: Slice n cases of a log
- `slice(grouped_log)`: Slice grouped log: take slice of cases from each group.

Functions

- `slice_activities(eventlog)`: Take a slice of activity instances from event log
- `slice_activities(activitylog)`: Take a slice of activity instances from activity log
- `slice_activities(grouped_log)`: Take a slice of activity instances from grouped event log
- `slice_events(eventlog)`: Take a slice of events from event log
- `slice_events(grouped_eventlog)`: Take a slice of events from grouped event log

`slice_activities` *Slice Activities*

Description

Take a slice of activity instances from event log

Usage

```
slice_activities(.data, ...)
```

Arguments

`.data`	`log`: Object of class `eventlog` or `activitylog`.
`...`	Slice index

`slice_events` *Slice Events*

Description

Take a slice of events from event log

Usage

```
slice_events(.data, ...)
```

Arguments

`.data`	`log`: Object of class `eventlog` or `activitylog`
`...`	Slice index

slice_sample *Sample function for logs*

Description

Sample function for logs

Usage

```
slice_sample(.data, ..., n, prop, by = NULL, weight_by = NULL, replace = FALSE)

## S3 method for class 'log'
slice_sample(.data, ..., n, prop, weight_by = NULL, replace = FALSE)

## S3 method for class 'grouped_log'
slice_sample(.data, ..., n, prop, weight_by = NULL, replace = FALSE)
```

Arguments

.data	A data frame, data frame extension (e.g. a tibble), or a lazy data frame (e.g. from dbplyr or dtplyr). See *Methods*, below, for more details.
...	Arguments passed on to dplyr::slice_sample n,prop Provide either n, the number of rows, or prop, the proportion of rows to select. If neither are supplied, n = 1 will be used. If n is greater than the number of rows in the group (or prop > 1), the result will be silently truncated to the group size. prop will be rounded towards zero to generate an integer number of rows. A negative value of n or prop will be subtracted from the group size. For example, n = -2 with a group of 5 rows will select 5 - 2 = 3 rows; prop = -0.25 with 8 rows will select 8 * (1 - 0.25) = 6 rows.
n, prop	Provide either n, the number of rows, or prop, the proportion of rows to select. If neither are supplied, n = 1 will be used. If n is greater than the number of rows in the group (or prop > 1), the result will be silently truncated to the group size. prop will be rounded towards zero to generate an integer number of rows. A negative value of n or prop will be subtracted from the group size. For example, n = -2 with a group of 5 rows will select 5 - 2 = 3 rows; prop = -0.25 with 8 rows will select 8 * (1 - 0.25) = 6 rows.
by	**[Experimental]** <tidy-select> Optionally, a selection of columns to group by for just this operation, functioning as an alternative to group_by(). For details and examples, see ?dplyr_by.
weight_by	<data-masking> Sampling weights. This must evaluate to a vector of non-negative numbers the same length as the input. Weights are automatically standardised to sum to 1.
replace	Should sampling be performed with (TRUE) or without (FALSE, the default) replacement.

Methods (by class)

- `slice_sample(log)`: Sample `n` cases of a `log`.
- `slice_sample(grouped_log)`: Sample `n` cases from a `grouped_log`.

`standardize_lifecycle` *Standardize format of lifecycle types*

Description

Standardize format of lifecycle types

Usage

```
standardize_lifecycle(eventlog)

## S3 method for class 'eventlog'
standardize_lifecycle(eventlog)
```

Arguments

`eventlog`	The event log to be converted. An object of class `eventlog`.

Methods (by class)

- `standardize_lifecycle(eventlog)`: Standardize lifecycle types for eventlog

`summarize` *Summarize event log*

Description

Summarize event log

Usage

```
summarise(.data, ..., .by = NULL, .groups = NULL)
```

Arguments

.data	log: Object of class eventlog or activitylog
...	Name-value pairs of summary functions
.by	**[Experimental]** <tidy-select> Optionally, a selection of columns to group by for just this operation, functioning as an alternative to group_by(). For details and examples, see ?dplyr_by.
.groups	**[Experimental]** Grouping structure of the result. • "drop_last": dropping the last level of grouping. This was the only supported option before version 1.0.0. • "drop": All levels of grouping are dropped. • "keep": Same grouping structure as .data. • "rowwise": Each row is its own group. When .groups is not specified, it is chosen based on the number of rows of the results: • If all the results have 1 row, you get "drop_last". • If the number of rows varies, you get "keep" (note that returning a variable number of rows was deprecated in favor of reframe(), which also unconditionally drops all levels of grouping). In addition, a message informs you of that choice, unless the result is ungrouped, the option "dplyr.summarise.inform" is set to FALSE, or when summarise() is called from a function in a package.

Value

The summarize function returns a tibble, no event log. All groups will be removed.

---

summary.eventlog *Generic summary function for eventlog class*

Description

Generic summary function for eventlog class

Usage

```
## S3 method for class 'eventlog'
summary(object, ...)

## S3 method for class 'grouped_eventlog'
summary(object, ...)
```

Arguments

object	log: Object of class eventlog or activitylog
...	Ignored.

Methods (by class)

- summary(grouped_eventlog): Summary of grouped event log

---

timestamp *Timestamp classifier*

Description

Get the timestamp classifier of an object of class eventlog

Usage

```
timestamp(x)

## S3 method for class 'eventlog'
timestamp(x)

## S3 method for class 'eventlog_mapping'
timestamp(x)

## S3 method for class 'activitylog'
timestamp(x)

## S3 method for class 'activitylog_mapping'
timestamp(x)
```

Arguments

x	Object of class eventlog, or mapping.

Methods (by class)

- timestamp(eventlog): Retrieve timestamp identifier from eventlog
- timestamp(eventlog_mapping): Retrieve timestamp identifier from eventlog mapping
- timestamp(activitylog): Retrieve timestamp identifier from activitylog
- timestamp(activitylog_mapping): Retrieve timestamp identifier from activitylog mapping

See Also

eventlog, mapping

Other Classifiers: activity_id(), activity_instance_id(), case_id(), lifecycle_id(), resource_id(), set_activity_id(), set_activity_instance_id(), set_case_id(), set_lifecycle_id(), set_resource_id(), set_timestamp(), timestamps()

---

timestamps *Timestamp classifiers*

Description

Get the timestamps classifier of an object of class activitylog

Usage

```
timestamps(x)

## S3 method for class 'eventlog'
timestamps(x)

## S3 method for class 'eventlog_mapping'
timestamps(x)

## S3 method for class 'activitylog'
timestamps(x)

## S3 method for class 'activitylog_mapping'
timestamps(x)
```

Arguments

x	Object of class activitylog, or mapping.

Methods (by class)

- timestamps(eventlog): Retrieve timestamp identifier from eventlog
- timestamps(eventlog_mapping): Retrieve timestamp identifier from eventlog mapping
- timestamps(activitylog): Retrieve timestamp identifier from activitylog
- timestamps(activitylog_mapping): Retrieve timestamp identifier from activitylog mapping

See Also

activitylog, mapping

Other Classifiers: activity_id(), activity_instance_id(), case_id(), lifecycle_id(), resource_id(), set_activity_id(), set_activity_instance_id(), set_case_id(), set_lifecycle_id(), set_resource_id(), set_timestamp(), timestamp()

to_activitylog *Convert eventlog object to activitylog object.*

Description

Convert eventlog object to activitylog object.

Usage

```
to_activitylog(eventlog)
```

Arguments

eventlog	Object of class eventlog

to_eventlog *Convert activitylog to eventlog*

Description

Convert activitylog to eventlog

Usage

```
to_eventlog(activitylog)

## S3 method for class 'activitylog'
to_eventlog(activitylog)

## S3 method for class 'grouped_activitylog'
to_eventlog(activitylog)
```

Arguments

activitylog	Object of class activitylog

Methods (by class)

- to_eventlog(activitylog): Convert activitylog to eventlog
- to_eventlog(grouped_activitylog): Convert grouped activitylog to grouped eventlog

traces *Traces*

Description

traces computes the different activity sequences of an event log together with their absolute and relative frequencies. Activity sequences are based on the start timestamp of activities.

Usage

```
traces(log, ...)

## S3 method for class 'log'
traces(log, ...)

## S3 method for class 'grouped_log'
traces(log, ...)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
...	Deprecated arguments

Methods (by class)

- traces(log): Construct traces list for eventlog
- traces(grouped_log): Construct list of traces for grouped log

See Also

cases, eventlog

trace_list *Trace list*

Description

Construct trace list

Usage

```
trace_list(log, ...)

## S3 method for class 'eventlog'
trace_list(log, ...)

## S3 method for class 'activitylog'
trace_list(log, ...)

## S3 method for class 'grouped_log'
trace_list(log, ...)
```

Arguments

log	log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
...	Other arguments. Currently not used.

Methods (by class)

- trace_list(eventlog): Construct trace list for event log
- trace_list(activitylog): Construct trace list for activity log
- trace_list(grouped_log): Construct list of traces for grouped log

---

ungroup *Ungroup log*

Description

Ungroup log

Usage

```
ungroup(x, ...)

## S3 method for class 'activitylog'
ungroup(x, ...)
```

Arguments

x	Activitylog
...	variables to remove from grouping

Methods (by class)

- ungroup(activitylog): Ungroup columns in eventlog

ungroup_eventlog *Ungroup event log*

Description

Remove groups from event log

Usage

```
ungroup_eventlog(log)

## S3 method for class 'eventlog'
ungroup_eventlog(log)

## S3 method for class 'grouped_log'
ungroup_eventlog(log)
```

Arguments

log	Eventlog

Methods (by class)

- ungroup_eventlog(eventlog): Remove groups from event log
- ungroup_eventlog(grouped_log): Remove groups from log.

unite *Unite multiple columns into one.*

Description

Unite multiple columns into one.

Usage

```
unite(data, col, ..., sep = "_", remove = TRUE, na.rm = FALSE)

## S3 method for class 'eventlog'
unite(data, col, ..., sep = "_", remove = T)

## S3 method for class 'activtylog'
unite(data, col, ..., sep = "_", remove = T)

## S3 method for class 'grouped_eventlog'
unite(data, col, ..., sep = "_", remove = T)
```

Arguments

data	Eventlog
col	The name of the new column, as a string or symbol. This argument is passed by expression and supports quasiquotation (you can unquote strings and symbols). The name is captured from the expression with rlang::ensym() (note that this kind of interface where symbols do not represent actual objects is now discouraged in the tidyverse; we support it here for backward compatibility).
...	Additional arguments passed to tidyr
sep	Separator to use between values.
remove	If TRUE, remove input columns from output data frame.
na.rm	If TRUE, missing values will be removed prior to uniting each value.

Methods (by class)

- unite(eventlog): Unite columns in eventlog
- unite(activtylog): Unite columns in activitylog
- unite(grouped_eventlog): Unite columns in grouped eventlog

Index