Package ‘bupaR’

April 3, 2023

**Type** Package

**Title** Business Process Analysis in R

**Version** 0.5.3

**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** 7.2.3

**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


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

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Gert Janssenswillen [aut, cre],
Gerard van Hulzen [ctb],
Felix Mannhardt [ctb],
Niels Martin [ctb],
Greg Van Houdt [ctb]

**Maintainer** Gert Janssenswillen <gert.janssenswillen@uhasselt.be>

**Repository** CRAN

**Date/Publication** 2023-04-02 22:00:06 UTC
<table>
<thead>
<tr>
<th>R topics documented:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>activities</td>
<td>3</td>
</tr>
<tr>
<td>activitylog</td>
<td>4</td>
</tr>
<tr>
<td>activity_id</td>
<td>5</td>
</tr>
<tr>
<td>activity_instance_id</td>
<td>5</td>
</tr>
<tr>
<td>activity_labels</td>
<td>6</td>
</tr>
<tr>
<td>act Collapse</td>
<td>7</td>
</tr>
<tr>
<td>act_recode</td>
<td>8</td>
</tr>
<tr>
<td>act_unite</td>
<td>9</td>
</tr>
<tr>
<td>add_end_activity</td>
<td>10</td>
</tr>
<tr>
<td>as.grouped.data.frame</td>
<td>11</td>
</tr>
<tr>
<td>assign_instance_id</td>
<td>11</td>
</tr>
<tr>
<td>bupaR</td>
<td>12</td>
</tr>
<tr>
<td>cases</td>
<td>12</td>
</tr>
<tr>
<td>case_id</td>
<td>13</td>
</tr>
<tr>
<td>case_labels</td>
<td>14</td>
</tr>
<tr>
<td>case_list</td>
<td>14</td>
</tr>
<tr>
<td>convert_timestamps</td>
<td>15</td>
</tr>
<tr>
<td>detect_resource_inconsistencies</td>
<td>16</td>
</tr>
<tr>
<td>durations</td>
<td>16</td>
</tr>
<tr>
<td>eventlog</td>
<td>17</td>
</tr>
<tr>
<td>events_to_activitylog</td>
<td>18</td>
</tr>
<tr>
<td>first_n</td>
<td>19</td>
</tr>
<tr>
<td>fix_resource_inconsistencies</td>
<td>20</td>
</tr>
<tr>
<td>grouped_activitylog</td>
<td>21</td>
</tr>
<tr>
<td>grouped_eventlog</td>
<td>21</td>
</tr>
<tr>
<td>grouped_log</td>
<td>21</td>
</tr>
<tr>
<td>group_by_activity</td>
<td>21</td>
</tr>
<tr>
<td>group_by_activity_instance</td>
<td>22</td>
</tr>
<tr>
<td>group_by_case</td>
<td>22</td>
</tr>
<tr>
<td>group_by_ids</td>
<td>23</td>
</tr>
<tr>
<td>group_by_resource</td>
<td>23</td>
</tr>
<tr>
<td>group_by_resource_activity</td>
<td>24</td>
</tr>
<tr>
<td>is.log</td>
<td>24</td>
</tr>
<tr>
<td>last_n</td>
<td>25</td>
</tr>
<tr>
<td>lifecycles</td>
<td>26</td>
</tr>
<tr>
<td>lifecycle_id</td>
<td>27</td>
</tr>
<tr>
<td>lifecycle_labels</td>
<td>27</td>
</tr>
<tr>
<td>log</td>
<td>28</td>
</tr>
<tr>
<td>mapping</td>
<td>28</td>
</tr>
<tr>
<td>n_activities</td>
<td>29</td>
</tr>
<tr>
<td>n_activity_instances</td>
<td>30</td>
</tr>
<tr>
<td>n_cases</td>
<td>31</td>
</tr>
<tr>
<td>n_events</td>
<td>31</td>
</tr>
<tr>
<td>n_resources</td>
<td>32</td>
</tr>
<tr>
<td>n_traces</td>
<td>33</td>
</tr>
<tr>
<td>print.eventlog_mapping</td>
<td>34</td>
</tr>
</tbody>
</table>
**activities**

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

**Usage**

```r
activities(log, ..., eventlog)
```

```r
## S3 method for class 'activitylog'
activities(log, ..., eventlog = deprecated())
```

```r
## S3 method for class 'grouped_log'
activities(log, ..., eventlog = deprecated())
```
Arguments

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

... Unused.

eventlog [Deprecated]: please use log instead.

Methods (by class)

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

See Also

activity_id, activity_instance_id, eventlog

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 referring 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", "manualskip", etc. 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

<table>
<thead>
<tr>
<th>activity_id</th>
<th>Activity classifier</th>
</tr>
</thead>
</table>

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(), timestamps(), timestamp()
activity_labels

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 of 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(),
timestamps(), timestamp()

activity_labels  
Get vector of activity labels

Description

Retrieve a vector containing all unique activity labels

Usage

activity_labels(log, eventlog = deprecated())

## S3 method for class 'log'
activity_labels(log, eventlog = deprecated())
**actCollapse**

**Arguments**

- **log**: Object of class `log` or derivatives (`grouped_log`, `eventlog`, `activitylog`, etc.).
- **eventlog** [Deprecated]: please use `log` instead.

**Methods (by class)**

- `activity_labels(log)`: Retrieve activity labels

---

**Description**

Collapse activity labels of a sub process into a single activity

**Usage**

```r
act Collapse(log, ..., method, eventlog = deprecated())
```

```r
## S3 method for class 'eventlog'
act Collapse(
  log,
  ..., 
  method = c("entry_points", "consecutive"),
  eventlog = deprecated()
)
```

```r
## S3 method for class 'activitylog'
act Collapse(
  log,
  ..., 
  method = c("entry_points", "consecutive"),
  eventlog = deprecated()
)
```

```r
## S3 method for class 'grouped_log'
act Collapse(
  log,
  ..., 
  method = c("entry_points", "consecutive"),
  eventlog = deprecated()
)
```
Arguments

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.

eventlog: [Deprecated]: please use log instead.

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 subprocess, by looking at the first and last activity in each case over the whole log. Subsequently, it will create a new instance of the subprocess 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()
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.

eventlog [Deprecated]: please use log instead.

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, ..., eventlog = deprecated())

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

## S3 method for class 'grouped_log'
act_unite(log, ..., eventlog = deprecated())

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.

eventlog [Deprecated]: please use log instead.
add_end_activity

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: actCollapse(), act_recode()

Description

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

Usage

```r
add_end_activity(log, label = "End", eventlog = deprecated())
```

```r
add_start_activity(log, label = "Start", eventlog = deprecated())
```

## S3 method for class 'eventlog'

```r
add_end_activity(log, label = "End", eventlog = deprecated())
```

## S3 method for class 'activitylog'

```r
add_end_activity(log, label = "End", eventlog = deprecated())
```

## S3 method for class 'grouped_log'

```r
add_end_activity(log, label = "End", eventlog = deprecated())
```

## S3 method for class 'eventlog'

```r
add_start_activity(log, label = "Start", eventlog = deprecated())
```

## S3 method for class 'activitylog'

```r
add_start_activity(log, label = "Start", eventlog = deprecated())
```

## S3 method for class 'grouped_log'

```r
add_start_activity(log, label = "Start", eventlog = deprecated())
```

Arguments

- `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`.
- `eventlog`: [Deprecated]: please use `log` instead.
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`.

---

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
timeline_id: Lifecycle identifier
See Also

Other Eventlog construction helpers: `convert_timestamps()`

---

**bupaR**

*bupaR - Business Process Analysis in R*

---

**cases**

*Cases*

---

**Description**

Functionalities for process analysis in R. This package 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.

---

**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**

```r
cases(log, ..., eventlog = deprecated())
```

### S3 method for class 'log'

```r
cases(log, ..., eventlog = deprecated())
```

### S3 method for class 'eventlog'

```r
cases(log, ..., eventlog = deprecated())
```

### S3 method for class 'activitylog'

```r
cases(log, ..., eventlog = deprecated())
```

**Arguments**

- **log**: Object of class `log` or derivatives (`grouped_log`, `eventlog`, `activitylog`, etc.).
- **...**: Other (optional) arguments passed on to methods. See `durations` for more options.
- **eventlog**: [Deprecated]: please use `log` instead.
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 an 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(), timestamps(), timestamp()

---

case_labels  Case labels

Description

Retrieve a vector containing all unique case labels

Usage

case_labels(log, eventlog = deprecated())

## S3 method for class 'log'
case_labels(log, eventlog = deprecated())

Arguments

log log: Object of class log or derivatives (grouped_log, eventlog, activitylog,
etc.).
eventlog [Deprecated]: please use log instead.

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, eventlog = deprecated())

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

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

Arguments

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.

.eventlog [Deprecated]: please use log instead.

Methods (by class)

• case_list(eventlog): Return case list
• case_list(activitylog): Return case list

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()
detect_resource_inconsistencies

*Detect resource inconsistencies*

**Description**

Function to detect inconsistencies in resource information between related events.

**Usage**

\[
\text{detect_resource_inconsistencies}(\text{eventlog, filter\_condition})
\]

**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**

\[
\text{durations}(
\begin{align*}
\text{log,} \\
\text{units = c("auto", "secs", "mins", "hours", "days", "weeks"),} \\
\text{eventlog = deprecated()}
\end{align*}
\)

**Arguments**

- `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`: [Deprecated]; please use `log` instead.
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**

```r
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

```r
## 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

```r
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.
**first_n**

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 arguments, i.e. for fixing resource inconsistencies

---

**first_n**  **first_n**

**Description**

Select first n activity instances.

**Usage**

```r
first_n(log, n, eventlog = deprecated())
```

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

## S3 method for class 'activitylog'
`first_n(log, n, eventlog = deprecated())`

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

**Arguments**

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

- **n**  integer: The number of activity instances to select.

- **eventlog**  [Deprecated]; please use log instead.

**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.
Description

Fix resource inconsistencies

Usage

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

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

```r
## 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_inconsistencies 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_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**

```r
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**

```r
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**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>log</td>
<td>Object of class <code>log</code> or derivatives (<code>grouped_log</code>, <code>eventlog</code>, <code>activitylog</code>, etc.).</td>
</tr>
<tr>
<td>...</td>
<td>One or more of the following: activity_id, case_id, activity_instance_id, resource_id, lifecycle_id</td>
</tr>
</tbody>
</table>

**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**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>log</td>
<td>Object of class <code>log</code> or derivatives (<code>grouped_log</code>, <code>eventlog</code>, <code>activitylog</code>, etc.).</td>
</tr>
</tbody>
</table>
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.).

---

**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.activitylog 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

Description

Select last n activity instances

Usage

last_n(log, n, eventlog = deprecated())

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

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

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

Arguments

log log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
n integer: The number of activity instances to select.
eventlog [Deprecated]: please use log instead.

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

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, eventlog = deprecated())

## S3 method for class 'eventlog'
lifecycles(log, eventlog = deprecated())

## S3 method for class 'grouped_eventlog'
lifecycles(log, eventlog = deprecated())

## S3 method for class 'activitylog'
lifecycles(log, eventlog = deprecated())

## S3 method for class 'grouped_activitylog'
lifecycles(log, eventlog = deprecated())

Arguments

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

eventlog [Deprecated]: please use log instead.

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**

```r
lifecycle_id(x)
```

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

```r
# 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()`, `timestamps()`, `timestamp()`

lifecycle_labels

----------------------------------  Get vector of lifecycle labels.
----------------------------------

**Description**

Retrieve a vector containing all unique lifecycle labels.

**Usage**

```r
lifecycle_labels(log, eventlog = deprecated())
```

```r
# S3 method for class 'eventlog'
lifecycle_labels(log, eventlog = deprecated())
```

```r
# S3 method for class 'activitylog'
lifecycle_labels(log, eventlog = deprecated())
```
Arguments

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

`eventlog` [Deprecated]: please use `log` instead.

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`

---

### mapping

**Arguments**

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

`eventlog` [Deprecated]: please use `log` instead.

**Description**

Prints the mapping of an event log object.

**Usage**

```r
mapping(log, eventlog = deprecated())
```

```r
## S3 method for class 'eventlog'
mapping(log, eventlog = deprecated())
```

```r
## S3 method for class 'activitylog'
mapping(log, eventlog = deprecated())
```

**Arguments**

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

`eventlog` [Deprecated]: please use `log` instead.
**n_activities**

**Methods (by class)**

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

**Description**

Returns the number of activities in an event log

**Usage**

```r
n_activities(log, eventlog = deprecated())
```

```r
## S3 method for class 'log'
n_activities(log, eventlog = deprecated())

## S3 method for class 'grouped_log'
n_activities(log, eventlog = deprecated())
```

**Arguments**

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

- `eventlog`  
  [Deprecated]: please use `log` instead.

**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

Description

Returns the number of activity instances in an event log

Usage

n_activity_instances(log, eventlog = deprecated())

## S3 method for class 'eventlog'
  n_activity_instances(log, eventlog = deprecated())

## S3 method for class 'grouped_eventlog'
  n_activity_instances(log, eventlog = deprecated())

## S3 method for class 'activitylog'
  n_activity_instances(log, eventlog = deprecated())

## S3 method for class 'grouped_activitylog'
  n_activity_instances(log, eventlog = deprecated())

Arguments

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

eventlog [Deprecated]: please use log instead.

Methods (by class)

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

See Also

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

<table>
<thead>
<tr>
<th>n_cases</th>
<th>n_cases</th>
</tr>
</thead>
</table>

**Description**

Returns the number of cases in an event log.

**Usage**

```r
n_cases(log, eventlog = deprecated())
```

```r
## S3 method for class 'log'
n_cases(log, eventlog = deprecated())
```

```r
## S3 method for class 'grouped_log'
n_cases(log, eventlog = deprecated())
```

**Arguments**

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

- `eventlog`  
  [Deprecated]: please use log instead.

**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**

<table>
<thead>
<tr>
<th>n_events</th>
<th>n_events</th>
</tr>
</thead>
</table>

**Description**

Returns the number of events in an event log.
Usage

\[ \text{n\_events}(\log, \text{eventlog} = \text{deprecated}()) \]

## S3 method for class 'eventlog'
\[ \text{n\_events}(\log, \text{eventlog} = \text{deprecated}()) \]

## S3 method for class 'grouped\_eventlog'
\[ \text{n\_events}(\log, \text{eventlog} = \text{deprecated}()) \]

## S3 method for class 'activitylog'
\[ \text{n\_events}(\log, \text{eventlog} = \text{deprecated}()) \]

## S3 method for class 'grouped\_activitylog'
\[ \text{n\_events}(\log, \text{eventlog} = \text{deprecated}()) \]

Arguments

log \hspace{1cm} \text{log}: \text{Object of class log or derivatives (grouped\_log, eventlog, activitylog, etc.).}

eventlog \hspace{1cm} \text{[Deprecated]}: \text{please use log instead.}

Methods (by class)

- \text{n\_events(eventlog)}: \text{Count number of events in an eventlog.}
- \text{n\_events(grouped\_eventlog)}: \text{Count number of events in a grouped\_eventlog.}
- \text{n\_events(activitylog)}: \text{Count number of events in an activitylog.}
- \text{n\_events(grouped\_activitylog)}: \text{Count number of events in a grouped\_activitylog.}

See Also

Other Counters: \text{n\_activities()}, \text{n\_activity\_instances()}, \text{n\_cases()}, \text{n\_resources()}, \text{n\_traces()}

<table>
<thead>
<tr>
<th>n_resources</th>
<th>n_resources</th>
</tr>
</thead>
</table>

Description

\text{Returns the number of resources in an event log}

Usage

\[ \text{n\_resources}(\log, \text{eventlog} = \text{deprecated}()) \]

## S3 method for class 'log'
\[ \text{n\_resources}(\log, \text{eventlog} = \text{deprecated}()) \]

## S3 method for class 'grouped\_log'
\[ \text{n\_resources}(\log, \text{eventlog} = \text{deprecated}()) \]
Arguments

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

eventlog: [Deprecated]: please use log instead.

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_resources()

Description

Returns the number of traces in an event log

Usage

n_traces(log, eventlog = deprecated())

## S3 method for class log
n_traces(log, eventlog = deprecated())

## S3 method for class grouped_log
n_traces(log, eventlog = deprecated())

Arguments

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

eventlog: [Deprecated]: please use log instead.

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
Resources

Description

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

Usage

resources(log, eventlog = deprecated())

## S3 method for class 'eventlog'
resources(log, eventlog = deprecated())

## S3 method for class 'activitylog'
resources(log, eventlog = deprecated())

## S3 method for class 'grouped_log'
resources(log, eventlog = deprecated())

Arguments

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

eventlog [Deprecated]: please use log instead.

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

<table>
<thead>
<tr>
<th>resource_id</th>
<th>Resource classifier</th>
</tr>
</thead>
</table>

Description

Get the resource classifier of an object of class `eventlog`.

Usage

```r
resource_id(x)
```

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

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

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

## S3 method for class 'activitylog_mapping'
```r
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()`, `timestamps()`, `timestamp()`
**resource_labels**

*Get vector of resource labels*

**Description**

Retrieve a vector containing all unique resource labels.

**Usage**

```r
resource_labels(log, eventlog = deprecated())
```

### Default S3 method:

```r
resource_labels(log, eventlog = deprecated())
```

**Arguments**

- `log`: Object of class `log` or derivatives (`grouped_log, eventlog, activitylog`, etc.).
- `eventlog`: [Deprecated]: please use `log` instead.

**Methods (by class)**

- `resource_labels(default)`: Retrieve resource labels from eventlog

---

**re_map**

*Re map*

**Description**

Construct an eventlog using an existing mapping.

**Usage**

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

**Arguments**

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

_bupaR color scales_

**Description**

_bupaR color scales_

**Usage**

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

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

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

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

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

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

```r
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.

---

**Description**

Select identifiers from log

**Usage**

```r
select_ids(log, ...)
```

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

**Arguments**

- `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, eventlog = deprecated())
```

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

Arguments

- `log`: Object of class `log` or derivatives (`grouped_log`, `eventlog`, `activitylog`, etc.).
- `activity_id`: New activity id
- `eventlog` [Deprecated]: please use `log` instead.

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()`, `timestamps()`, `timestamp()`
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, eventlog = deprecated())

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

Arguments
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(),
timestamps(), timestamp()

set_case_id

Set case id of log

Description
Set case id of log

Usage
set_case_id(log, case_id, eventlog = deprecated())

## Default S3 method:
set_case_id(log, case_id, eventlog = deprecated())

##
set_lifecycle_id

Arguments

log log: Object of class log or derivatives (grouped_log, eventlog, activitylog, etc.).
case_id New case id
eventlog [Deprecated]: please use log instead.

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(), timestamps(), timestamp()

set_lifecycle_id Set lifecycle id of log

Description

Set lifecycle id of log

Usage

set_lifecycle_id(log, lifecycle_id, eventlog = deprecated())

## Default S3 method:
set_lifecycle_id(log, lifecycle_id, eventlog = deprecated())

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
eventlog [Deprecated]: please use log instead.

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(),
timestamps(), timestamp()
### set_resource_id

**Description**

Set resource id of log

**Usage**

```r
set_resource_id(log, resource_id, eventlog = deprecated())
```

**Arguments**

- `log`: Object of class `log` or derivatives (`grouped_log`, `eventlog`, `activitylog`, etc.).
- `resource_id`: New resource id.
- `eventlog` [Deprecated]: please use `log` instead.

**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()`, `timestamps()`, `timestamp()`

### set_timestamp

**Description**

Set timestamp of eventlog

**Usage**

```r
set_timestamp(log, timestamp, eventlog = deprecated())
```

**Methods (by class)**

## Default S3 method: set_resource_id

```r
set_resource_id(log, resource_id, eventlog = deprecated())
```

**Arguments**

- `log`: Object of class `log` or derivatives (`grouped_log`, `eventlog`, `activitylog`, etc.).
- `resource_id`: New resource id.
- `eventlog` [Deprecated]: please use `log` instead.
Arguments

- **log**: Object of class `log` or derivatives (`grouped_log`, `eventlog`, `activitylog`, etc.).
- **timestamp**: New timestamp
- **eventlog**: `[Deprecated]`: please use `log` instead.

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()`, `timestamps()`, `timestamp()`

---

### 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**

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

isimple_eventlog(eventlog)
```
slice_activities

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventlog</td>
<td>The data object to be used as event log. This can be a data.frame or tibble.</td>
</tr>
<tr>
<td>case_id</td>
<td>The case classifier of the event log.</td>
</tr>
<tr>
<td>activity_id</td>
<td>The activity classifier of the event log.</td>
</tr>
<tr>
<td>timestamp</td>
<td>The timestamp of the event log.</td>
</tr>
<tr>
<td>resource_id</td>
<td>The resource classifier of the event log (optional).</td>
</tr>
<tr>
<td>order</td>
<td>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.</td>
</tr>
<tr>
<td>return_type</td>
<td>Whether to return eventlog (default) or activitylog object.</td>
</tr>
</tbody>
</table>

See Also

eventlog, case_id, activity_id, activity_instance_id, lifecycle_id, timestamp

Examples

```r
# 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_activities  

Slice Activities

Description

Take a slice of activity instances from event log

Usage

slice_activities(.data, ...)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.data</td>
<td>log: Object of class eventlog or activitylog.</td>
</tr>
<tr>
<td>...</td>
<td>Slice index</td>
</tr>
</tbody>
</table>
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

standardize_lifecycle  

Standardize format of lifecycle types

Description

Standardize format of lifecycle types

Usage

standardize_lifecycle(eventlog)

## S3 method for class 'eventlog'
estandardize_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
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)
timestamps

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

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, ..., eventlog = deprecated())

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

## S3 method for class 'grouped_log'
traces(log, ..., eventlog = deprecated())

Arguments

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

... Deprecated arguments

eventlog [Deprecated]: please use log instead.

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

Description

Construct trace list

Usage

trace_list(log, ..., eventlog = deprecated())

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

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

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

Arguments

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

... Other arguments. Currently not used.

eventlog [Deprecated]: please use log instead.

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_eventlog

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.
Index

* Activity processing functions
  actCollapse, 7
  actRecode, 8
  actUnite, 9

* Classifiers
  activity_id, 5
  activity_instance_id, 5
  case_id, 13
  lifecycle_id, 27
  resource_id, 36
  set_activity_id, 40
  set_activity_instance_id, 41
  set_case_id, 41
  set_lifecycle_id, 42
  set_resource_id, 43
  set_timestamp, 43
  timestamp, 47
  timestamps, 48

* Counters
  n_activities, 29
  n_activity_instances, 30
  n_cases, 31
  n_events, 31
  n_resources, 32
  n_traces, 33

* Eventlog classifiers
  mapping, 28

* Eventlog construction helpers
  assign_instance_id, 11
  convert_timestamps, 15

actCollapse, 7, 9, 10
actRecode, 8, 8, 10
actUnite, 8, 9, 9
activities, 3
activity_id, 4, 5, 6, 9, 10, 14, 18, 27, 36, 40–45, 48, 49
activity_instance_id, 4, 5, 5, 14, 18, 27, 36, 40–45, 48, 49
activity_labels, 6
activitylog, 4, 4, 5, 7–16, 19, 21–26, 28–37, 39–51
add_end_activity, 10
add_start_activity(add_end_activity), 10
as.grouped.data.frame, 11
assign_instance_id, 11, 15
bupaR, 12
case_id, 5, 6, 13, 18, 27, 36, 40–45, 48, 49
case_labels, 14
case_list, 13, 14
cases, 12, 50
character, 10, 16
convert_timestamps, 12, 15
detect_resource_inconsistencies, 16
difftime, 16
durations, 12, 13, 16
eventlog, 4, 5, 7–16, 17, 19, 21–37, 39–51
events_to_activitylog, 18
first_n, 19
fix_resource_inconsistencies, 20
group_by_activity, 21
group_by_activity_instance, 22
group_by_case, 22
group_by_ids, 23
group_by_resource, 23
group_by_resource_activity, 24
grouped_activitylog, 21, 25, 26, 32
grouped_eventlog, 21, 25, 26, 32
grouped_log, 4, 7–12, 14–16, 19, 21, 21, 22–26, 28–33, 35, 37, 40–44, 50, 51
ieventlog (eventlog), 17
integer, 19, 25
is.activitylog (is.log), 24