Package: tokr (via r-universe)

November 2, 2026

Type Package

Title Process Models in the Plain-Text Language 'tok'

Version 1.0.3

Description Reads, checks, formats and exports process models written in 'tok', a plain-text language for Business Process Model and Notation (BPMN) 2.0 processes with their Decision Model and Notation (DMN) decisions, 'Camunda' forms and simulation parameters. Models are checked against the rules of BPMN 2.0, with diagnostics in English or Polish, and exported to BPMN files with a diagram layout, Scalable Vector Graphics (SVG) pictures, DMN and form files; they are drawn in the viewer and in 'R Markdown' and 'Quarto' documents through a 'knitr' engine. Processes are simulated as discrete-event models, giving indicators, an event log for process mining and a table of cases. The work is done by the 'Rust' engine of 'tok', shared with its 'Python' package and command line tool.

License GPL (>= 3)

URL https://tok-lang-bpm.github.io/, https://github.com/tok-lang-bpm/tok

BugReports https://github.com/tok-lang-bpm/tok/issues

Encoding UTF-8

Depends R (>= 4.2)

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

SystemRequirements Cargo (Rust's package manager), rustc >= 1.88

Roxygen list(markdown = TRUE)

RoxygenNote 8.0.0

Copyright Marek Deja; the authors of the bundled Rust crates listed in inst/AUTHORS.

NeedsCompilation yes

Author Marek Deja [aut, cre, cph]

Maintainer Marek Deja <marek.deja@uj.edu.pl>

Config/pak/sysreqs libclang-dev

Repository https://tok-lang-bpm.r-universe.dev

Date/Publication 2026-10-31 15:09:10 UTC

RemoteUrl https://github.com/tok-lang-bpm/tokr

RemoteRef HEAD

RemoteSha 6e994d31d07fbc620c9818dafba900df087e8a96

Contents

eng_tok . . . 2
tok_bpmn . . . 3
tok_check . . . 4
tok_eval . . . 5
tok_export . . . 5
tok_fmt . . . 6
tok_import . . . 7
tok_plot . . . 7
tok_read . . . 8
tok_sim . . . 9
tok_version . . . 10
tok_xes . . . 11

Index **13**

`eng_tok` *knitr engine for tok*

Description

Loading the package registers the engine `tok`, so a chunk written as ```` ```{tok} ```` in R Markdown or Quarto holds a model. Chunk options:

Usage

```
eng_tok(options)
```

Arguments

`options`	Chunk options, passed by knitr.

Details

- `tok.output`: `"svg"` (default) draws the diagram; `"check"` prints the diagnostics; `"fmt"` prints the canonical text; `"bpmn"` prints the BPMN XML.
- `tok.title`: title drawn above the diagram.
- `tok.lang`: language of the messages (default: the option `tok.lang`, then `"en"`).
- `tok.export`: a directory; every file of `tok export` is also written there.

The diagram is written as an SVG file and included like a figure (`fig.cap` and the other figure options apply), which suits HTML output; for PDF output use `tok.output = "check"` or include an exported picture.

Value

The chunk output, for knitr.

Examples

```
# the engine is registered when the package is loaded
if (requireNamespace("knitr", quietly = TRUE)) {
  is.function(knitr::knit_engines$get("tok"))
}
```

---

tok_bpmn *BPMN 2.0 XML and SVG of a model*

Description

`tok_bpmn()` exports BPMN 2.0 XML with diagram layout, which opens in Camunda Modeler and bpmn.io; `tok_svg()` draws the diagram. Both stop with the list of errors when the model has errors (see `tok_check()`).

Usage

```
tok_bpmn(x, file = NULL, lang = getOption("tok.lang", "en"))

tok_svg(x, file = NULL, title = NULL, lang = getOption("tok.lang", "en"))
```

Arguments

`x`	A model: a `tok_model`, the path of a `.tok` file, or the text of a model.
`file`	Optional path to write to.
`lang`	Language of the messages, `"en"` or `"pl"`. The default comes from the option `tok.lang`.
`title`	Title drawn above the diagram (SVG only).

Value

The XML or SVG text; invisibly when written to `file`.

Examples

```
model <- system.file("examples", "wypozyczenie.tok", package = "tokr")
xml <- tok_bpmn(model)
substr(xml, 1, 120)
svg <- tok_svg(model, title = "Interlibrary loan")
nchar(svg)
```

`tok_check` *Check a model*

Description

Runs the checks of `tok check`: syntax, identifiers, flows, pools and lanes, decision tables, forms, simulation parameters and the BPMN 2.0 rules.

Usage

```
tok_check(
  x,
  lang = getOption("tok.lang", "en"),
  level = c("full", "analytic", "basic"),
  bpmn = TRUE
)
```

Arguments

`x`	A model: a `tok_model`, the path of a `.tok` file, or the text of a model.
`lang`	Language of the messages, `"en"` or `"pl"`. The default comes from the option `tok.lang`.
`level`	`"full"` (default), `"analytic"` or `"basic"`: the lower levels also report the use of statements that belong to a higher one.
`bpmn`	Whether to run the BPMN 2.0 rules on the exported model.

Value

A data frame of class `tok_diagnostics` with the columns `line`, `code`, `severity` (`"error"`, `"warning"` or `"note"`), `id` and `message`, one row per problem; no rows when the model has no problems.

Examples

```
tok_check('tok 0.3 "Example"
start s "Start"
task a "Work"
end e "End"
flow s -> a -> e')
```

tok_eval *Evaluate a decision*

Description

Evaluates a decision of the model for the given inputs, as `tok eval` does. In a model of language 0.7 the decision is evaluated with the decisions, input data and business knowledge models it requires, and the inputs are the input data by FEEL name; in earlier models the decision table alone is evaluated with the hit policy of the table.

Usage

```
tok_eval(x, decision, inputs = list(), lang = getOption("tok.lang", "en"))
```

Arguments

`x`	A model: a `tok_model`, the path of a `.tok` file, or the text of a model.
`decision`	Identifier of the decision.
`inputs`	Named list of inputs: numbers, strings, logical values, `NA` for null; a vector longer than one is a list, a named list is a context.
`lang`	Language of the messages, `"en"` or `"pl"`. The default comes from the option `tok.lang`.

Value

In a model of language 0.7 the value of the decision (a number, string, logical value, list or named list; `NULL` for null), a list of named lists as a data frame. In earlier models: for the hit policies `unique`, `first`, `any` and `priority` a named list output name → value (`NULL` when no rule matches); for `collect` a data frame with one row per matched rule.

Examples

```
f <- system.file("examples", "propozycja-zakupu.tok", package = "tokr")
tok_eval(f, "tryb-akceptacji", list(rodzaj = "drukowana", cena = 120))
```

tok_export *Export every file of a model*

Description

Writes what `tok export` writes: `NAME.bpmn`, `NAME.svg`, one `.dmn` file per decision table and one `.form` file per form, ready to deploy to Camunda 8.

Usage

```
tok_export(x, dir = ".", name = NULL, lang = getOption("tok.lang", "en"))
```

Arguments

x	A model: a tok_model, the path of a .tok file, or the text of a model.
dir	Output directory, created when missing.
name	Base name of the BPMN and SVG files; by default the name of the .tok file.
lang	Language of the messages, "en" or "pl". The default comes from the option tok.lang.

Value

The paths of the written files, invisibly.

Examples

```
out <- file.path(tempdir(), "tok-export")
files <- tok_export(system.file("examples", "propozycja-zakupu.tok", package = "tokr"),
                    dir = out)
basename(files)
unlink(out, recursive = TRUE)
```

---

tok_fmt *Canonical text of a model*

Description

Prints the model as tok fmt does: one statement per line, canonical order of attributes, comments kept.

Usage

```
tok_fmt(x, write = FALSE, lang = getOption("tok.lang", "en"))
```

Arguments

x	A model: a tok_model, the path of a .tok file, or the text of a model.
write	When TRUE and x is a file, the file is rewritten.
lang	Language of the messages, "en" or "pl". The default comes from the option tok.lang.

Value

The canonical text (invisibly when the file is rewritten).

Examples

```
tok_fmt(system.file("examples", "wypozyczenie.tok", package = "tokr"))
```

tok_import *Convert BPMN 2.0 to tok*

Description

Reads a BPMN 2.0 file (for example drawn in Camunda Modeler) and returns the model as tok text, as `tok import` does.

Usage

```
tok_import(file, keep_ids = TRUE)
```

Arguments

`file`	Path of a `.bpmn` file.
`keep_ids`	Keep the identifiers of the BPMN file; `FALSE` derives readable ones from the labels.

Value

A `tok_model`; warnings about elements tok does not keep are in its element `warnings` and are also signalled.

Examples

```
bpmn <- tempfile(fileext = ".bpmn")
tok_bpmn(system.file("examples", "wypozyczenie.tok", package = "tokr"), file = bpmn)
m <- tok_import(bpmn)
m
unlink(bpmn)
```

tok_plot *Draw a model*

Description

Shows the diagram in the RStudio or Positron viewer, or in the web browser.

Usage

```
tok_plot(x, title = NULL, lang = getOption("tok.lang", "en"))
```

Arguments

x	A model: a tok_model, the path of a .tok file, or the text of a model.
title	Title drawn above the diagram (SVG only).
lang	Language of the messages, "en" or "pl". The default comes from the option tok.lang.

Value

The path of the temporary SVG file, invisibly.

Examples

```
if (interactive()) {
  tok_plot(system.file("examples", "wypozyczenie.tok", package = "tokr"))
}
```

---

tok_read *Read a tok model*

Description

Reads a .tok file. The result can be passed to every other function of the package; they also accept the path of a .tok file or the text of a model directly.

Usage

```
tok_read(path)
```

Arguments

path	Path of a .tok file.

Value

An object of class tok_model: a list with the elements text, path and name (the file name without the extension).

Examples

```
m <- tok_read(system.file("examples", "wypozyczenie.tok", package = "tokr"))
m
```

tok_sim *Simulate a process*

Description

Runs the discrete-event simulation of `tok sim`: cases arrive, wait for the resources of `by=`, take the durations of `duration=` and choose flows by their conditions and probabilities, as the `simulate`, `resource` and `variable` statements of the model say. The same model and seed always give the same result, number by number the result of `tok sim` and of the Python package.

Usage

```
tok_sim(x, cases = NULL, seed = NULL, lang = getOption("tok.lang", "en"))
```

Arguments

`x`	A model: a `tok_model`, the path of a `.tok` file, or the text of a model.
`cases`	Number of cases; `NULL` takes `cases=` of `simulate` (default 100).
`seed`	Seed of the random number generator; `NULL` takes `seed=` of `simulate` (default 1). Like `tok sim`, the simulator takes the integer part of the number, so `seed = 7.9` is the seed 7.
`lang`	Language of the messages, `"en"` or `"pl"`. The default comes from the option `tok.lang`.

Details

Times of the simulation are calendar times counted from `start=` of `simulate`. In the event log and the table of cases they are `POSIXct` values in UTC; a `start=` without a UTC offset is taken as UTC, so the clock times are those of `tok sim --log`.

The event log has one row per activity instance, as `tok sim --log` writes it: the columns `case`, `activity` (identifier), `label`, `resource` (`""` when the activity has no `by=`), `enqueue` (when the case reached the activity), `start` (when a performer started the work; `NA` when the activity was interrupted while waiting), `end` (when it completed or was interrupted) and `status` (`"completed"` or `"interrupted"`). The columns `case_id`, `activity_id` (the label, or the identifier when the activity has no label), `activity_instance_id` (the row number), `resource_id` (`NA` when none), `schedule` (= `enqueue`) and `complete` (= `end` of completed instances, `NA` for interrupted ones) repeat them under the names process-mining packages expect. For example, with bupaR:

```
bupaR::activitylog(s$log, case_id = "case_id", activity_id = "activity_id",
                   resource_id = "resource_id",
                   timestamps = c("schedule", "start", "complete"))
```

Value

An object of class `tok_sim`, a list with the elements:

- `kpi`: the indicators of `tok sim --json`, a list with `seed`; `cases` (a list of `started`, `finished` and `by_end_event`, a named integer vector of finished cases per end event); `cycle_time` (a list of `mean_s`, `median_s` and `p90_s` in seconds, `NA` without finished cases, and the same as short texts `mean`, `median`, `p90`); `activities`, a data frame with one row per activity (`activity`, `label`, `count` of completed instances, `mean_wait_s`, `mean_processing_s`, `mean_wait`, `mean_processing`); `resources`, a data frame with one row per resource (`resource`, `label`, `count`, `utilization` as a fraction); `horizon_s`, the simulated period in seconds; and `warnings`.
- `log`: the event log, a data frame described above.
- `cases`: a data frame with one row per case: `case`, `start`, `end` (`NA` for an unfinished case), `end_event` (`NA` for an unfinished case), `cycle_time_s`, and one column per case variable. A variable that holds a single value in every case is a vector (logical, numeric or character, `NA` where the case has no value; durations in seconds), any other a list; a variable named like one of the first five columns gets the prefix `variable_`.
- `warnings`: conditions or expressions the simulator could not evaluate.
- `seed`: the seed used, as a number.
- `horizon`: the simulated period in seconds, from the first arrival to the last event.
- `report`: the text report of `tok sim` in the language `lang`, which `print()` shows.

The object also keeps the model, the number of cases and the seed of the run, so `tok_xes()` can write its event log as XES.

`as.data.frame()` returns the event log. The function stops with the errors of the model (see `tok_check()`) and with the errors of the simulator as `tok sim` reports them, for example `tok.sim.SimError: the process`

Examples

```
f <- system.file("examples", "kwerenda-symulacja.tok", package = "tokr")
s <- tok_sim(f)
s
s$kpi$resources
head(s$log)
table(tok_sim(f, cases = 50, seed = 3)$cases$end_event)
```

---

tok_version *Version of the tok engine*

Description

Version of the tok engine

Usage

```
tok_version()
```

Value

The version of the Rust engine compiled into the package.

Examples

```
tok_version()
```

`tok_xes` *XES event log of a simulation*

Description

Writes the event log of `tok_sim()` as an XES document (IEEE 1849-2016, eXtensible Event Stream), the exchange format of process-mining tools such as ProM, pm4py, bupaR, Disco and Celonis; it is the file of `tok sim --xes`.

Usage

```
tok_xes(
  x,
  file = NULL,
  cases = NULL,
  seed = NULL,
  lang = getOption("tok.lang", "en")
)
```

Arguments

`x`	A result of `tok_sim()`, whose run is repeated with the same model, number of cases and seed (which gives the same log), or a model: a `tok_model`, the path of a `.tok` file, or the text of a model, which is simulated with `cases` and `seed`.
`file`	Optional path to write to.
`cases, seed`	As in `tok_sim()`; only for a model, not for a result of `tok_sim()`.
`lang`	Language of the messages, `"en"` or `"pl"`. The default comes from the option `tok.lang`.

Details

The log has one trace per case and up to three events per activity instance, with the transitions of the standard lifecycle model: `schedule` when the case reached the activity, `start` when a performer started the work and `complete` when it completed; an instance interrupted by a boundary timer ends with `ati_abort` when it was running and with `withdraw` when it was still waiting for a performer. Events carry `concept:name` (the label of the activity), `concept:instance`, `org:resource`, `time:timestamp` (with milliseconds and the UTC offset of `start=` of `simulate`) and `activity` (the identifier). Traces carry the case number, `start`, `end`, `end_event`, `cycle_time_s` and the case variables as typed attributes. See `docs/simulation.md` in the repository for the details.

Value

The XES text; invisibly when written to `file`.

Examples

```
f <- system.file("examples", "kwerenda-symulacja.tok", package = "tokr")
s <- tok_sim(f, cases = 20)
xes <- tempfile(fileext = ".xes")
tok_xes(s, xes)
substr(tok_xes(f, cases = 2, seed = 7), 1, 200)
```

Index

eng_tok, 2

tok_bpmn, 3
tok_check, 4
tok_check(), *3*, *10*
tok_eval, 5
tok_export, 5
tok_fmt, 6
tok_import, 7
tok_plot, 7
tok_read, 8
tok_sim, 9
tok_sim(), *11*
tok_svg *(*tok_bpmn*)*, 3
tok_version, 10
tok_xes, 11
tok_xes(), *10*