Package: plan (via r-universe)

October 12, 2024

Type Package

Title Tools for Project Planning

Version 0.4-5

Author Dan Kelley <Dan.Kelley@Dal.Ca> [aut,cre], Frank Schmitt <frankschmitt@gmx.de> [aut]

Maintainer Dan Kelley <Dan.Kelley@Dal.Ca>

Depends R (>= 2.15), utils, methods

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Description Supports the creation of 'burndown' charts and 'gantt' diagrams.

License GPL (>= 2)

Encoding UTF-8

URL https://github.com/dankelley/plan, https://dankelley.github.io/plan/

LazyData false

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

VignetteBuilder knitr

Language en-US

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

RemoteUrl https://github.com/dankelley/plan

RemoteRef HEAD

RemoteSha c60d22083727824abef6ae1708e6c4c17e4e78ed

Contents

`as.burndown` *Create a burndown object*

Description

Create a burndown object from the given data.

Usage

```
as.burndown(start, deadline, tasks, progress, progressInPercent = FALSE)
```

Arguments

`start`	Start date
`deadline`	Deadline (end date)
`tasks`	Data frame containing the task IDs (may be alphanumeric), their description and effort
`progress`	Data frame containing the progress values with task ID, timestamp and work done (either in percentage or absolute)
`progressInPercent`	boolean; if set to `FALSE`, progress values are treated like absolute values and converted to percentages

Details

Creates a burndown object from the given data; progress may be given in percentage or absolute values.

Value

A burndown object.

Author(s)

Frank Schmitt

See Also

Other things related to burndown data: burndown-class, burndown, plot,burndown-method, read.burndown(), summary,burndown-method

Examples

```
library(plan)
# same data as in tests/burndown.dat
start <- as.POSIXct(strptime("2006-04-08 12:00:00", "%Y-%m-%d %H:%M:%S"))
deadline <- as.POSIXct(strptime("2006-04-11 20:00:00", "%Y-%m-%d %H:%M:%S"))
tasks <- data.frame(key = c(1, 2, 3, 4, 5, 6),
    description = c("code read.burndown()", "code summary.burndown()",
        "code plot.burndown()", "create R package",
        "write documentation", "set up website"),
    effort = c(4, 1, 5, 2, 2, 1),
    stringsAsFactors = FALSE)
progress <- data.frame(key = c(1, 2, 1, 2, 4, 5, 4, 1, 3, 3, 3, 2, 2, 1, 5, 5, 5, 1, 3, 6),
    progress = c(5, 5, 10, 50, 5, 5, 100, 50, 5, 30, 80, 60,
        100, 70, 30, 90, 100, 100, 100, 100),
    time = structure(c(1144494000, 1144495800, 1144497600, 1144501200,
            1144517400, 1144519200, 1144523760, 1144566600,
            1144568460, 1144570680, 1144573200, 1144576800,
            1144577400, 1144578600, 1144583400, 1144585200,
            1144585800, 1144586100, 1144586400, 1144591200),
        class = "POSIXct"),
    stringsAsFactors = FALSE
)
b <- as.burndown(start, deadline, tasks, progress, progressInPercent = TRUE)
summary(b)
plot(b)
```

`as.gantt` *Create a gantt object.*

Description

This creates a gantt object.

Usage

```
as.gantt(key, description, start, end, done, neededBy)
```

Arguments

`key`	integer key for task, normally 1 for the first task, 2 for the second, etc.
`description`	character string describing the task (brief)
`start`	start date for task (POSIXt or character string that converts to POSIXt with `as.POSIXct()`
`end`	end date for task (POSIXt or character string that converts to POSIXt with `as.POSIXct()`.
`done`	percentage completion for the task
`neededBy`	optional key for a dependent task

Value

A gantt object; for details, see `read.gantt()`.

Author(s)

Dan Kelley

See Also

Other things related to gantt data: `gantt-class`, `ganttAddTask()`, `gantt`, `plot,gantt-method`, `read.gantt()`, `summary,gantt-method`

Examples

```
library(plan)
arrive <- as.POSIXct("2012-09-05")
month <- 28 * 86400
year <- 12 * month
leave <- arrive + 4 * year
startT1 <- arrive
endT1 <- startT1 + 4 * month
startT2 <- endT1 + 1
endT2 <- startT2 + 4 * month
startQE <- arrive + 9 * month
```

```
endQE <- arrive + 12 * month
QEabsoluteEnd <- arrive + 15 * month
startProposal <- arrive + 15 * month # for example
endProposal <- arrive + 20 * month
startThesisWork <- arrive + 2 * month # assumes no thesis work until 2 months in
endThesisWork <- leave - 4 * month
startWriting <- leave - 36 * month
endWriting <- leave
g <- as.gantt(key=1:8, c("Academic",
              "Term 1 classes",
              "Term 2 classes",
              "Qualifying Examination",
              "Research",
              "Proposal Defence",
              "Thesis Work",
              "Paper/Thesis Writing"),
              c(startT1, startT1, startT2, startQE, startProposal, startProposal,
                startThesisWork, startWriting),
              c(startT1, endT1, endT2, endQE, startProposal, endProposal,
                endThesisWork, endWriting),
              done=rep(0, 7))
plot(g, xlim=c(arrive, leave),
     ylabel=list(font=c(2,rep(1,3),2), justification=c(0,rep(1,3),0)))
```

---

burndown *Sample burndown dataset*

Description

This is sample burndown dataset provided for testing.

Author(s)

Dan Kelley

See Also

Other things related to burndown data: `as.burndown()`, `burndown-class`, `plot,burndown-method`, `read.burndown()`, `summary,burndown-method`

Other data sets provided with plan: `gantt`

burndown-class *Class to store burndown objects*

Description

Class to store burndown objects

See Also

Other things related to burndown data: `as.burndown()`, `burndown`, `plot,burndown-method`, `read.burndown()`, `summary,burndown-method`

check.tokens *check tokens*

Description

check tokens

Usage

```
check.tokens(tokens, expected)
```

Arguments

`tokens`	the tokens
`expected`	as expected

gantt *Sample gantt dataset*

Description

This is sample gantt dataset provided for testing.

Author(s)

Dan Kelley

See Also

Other things related to gantt data: `as.gantt()`, `gantt-class`, `ganttAddTask()`, `plot,gantt-method`, `read.gantt()`, `summary,gantt-method`

Other data sets provided with plan: `burndown`

gantt-class *Class to store gantt objects*

Description

These objects may be created with `as.gantt()` or `read.gantt()`.

See Also

Other things related to gantt data: `as.gantt()`, `ganttAddTask()`, `gantt`, `plot,gantt-method`, `read.gantt()`, `summary,gantt-method`

ganttAddTask *Add a task to a gantt object*

Description

This can be a simpler method than using `as.gantt()`, because tasks can be added one at a time.

Usage

```
ganttAddTask(
  g,
  description = "",
  start = NA,
  end = NA,
  done = 0,
  neededBy = NA,
  key
)
```

Arguments

g	A gantt object.
description	A character string describing the task.
start	A character string indicating the task start time, in a format understood by `as.POSIXct()`. Set to `""` (the default) to indicate that `description` is a heading, with no start and end time.
end	A character string indicating the end time, in a format understood by `as.POSIXct()`.
done	A numerical value indicating the fraction done.
neededBy	An integer indicating a task that depends on the completion of this task. If this is `NA`, then the task is not needed by any other task.

key — An optional value indicating the desired key value. If not given, this will default to one beyond the highest key in g. Otherwise, if key is an integer matching a task that is already in g, then that task is replaced; otherwise, the new task is placed between the tasks with integral keys on either side of the task. For example, setting key=4.5 places this between existing keys 4 and 5 (and then renumbers all keys to be integers); see "Examples".

See Also

Other things related to gantt data: as.gantt(), gantt-class, gantt, plot,gantt-method, read.gantt(), summary,gantt-method

Examples

```
library("plan")
g <- new("gantt")
g <- ganttAddTask(g, "Courses") # no times, so a heading
g <- ganttAddTask(g, "Physical Oceanography", "2016-09-03", "2016-12-05")
g <- ganttAddTask(g, "Chemistry Oceanography", "2016-09-03", "2016-12-05")
g <- ganttAddTask(g, "Fluid Dynamics", "2016-09-03", "2016-12-05")
g <- ganttAddTask(g, "Biological Oceanography", "2017-01-03", "2017-04-05")
g <- ganttAddTask(g, "Geological Oceanography", "2017-01-03", "2017-04-05")
g <- ganttAddTask(g, "Time-series Analysis", "2017-01-03", "2017-04-05")
g <- ganttAddTask(g, "Research") # no times, so a heading
g <- ganttAddTask(g, "Literature review", "2016-09-03", "2017-04-05")
g <- ganttAddTask(g, "Develop analysis skills", "2016-09-03", "2017-08-01")
g <- ganttAddTask(g, "Thesis work", "2017-01-01", "2018-04-01")
g <- ganttAddTask(g, "Defend thesis proposal", "2017-05-01", "2017-06-01")
g <- ganttAddTask(g, "Write papers & thesis", "2017-05-01", "2018-04-01")
g <- ganttAddTask(g, "Defend thesis", "2018-05-01", "2018-05-15")
# Set 'font' for bold-faced headings
font <- ifelse(is.na(g[["start"]]), 2, 1)
plot(g, ylabel=list(font=font))
```

---

plan-class *Base Class for plan Objects*

Description

Base Class for plan Objects

Slots

data A list containing variable contents.

plot,burndown-method *Draw a burndown chart*

Description

Plot a burndown chart.

Usage

```
## S4 method for signature 'burndown'
plot(
  x,
  col = NULL,
  draw.plan = TRUE,
  draw.regression = TRUE,
  draw.lastupdate = FALSE,
  t.stop = "",
  y.name = "Remaining Effort",
  debug = FALSE,
  ...
)
```

Arguments

x	A burndown object.
col	list of colours for items, starting with the first key in the file (which will be at the bottom of the chart). If not specified, the hcl() scheme will be used, to generate colours that are distinct, that show up reasonably well on a monitor.
draw.plan	boolean, set to TRUE to draw the plan, as a blue descending line with a horizontal intercept.
draw.regression	boolean, set to TRUE to draw a red dashed line indicating the overall progress, as determined by regression.
draw.lastupdate	boolean, set to TRUE to draw the last update (which otherwise requires a sharp eye).
t.stop	a POSIX time, the maximum time for graph (defaults to deadline if not given).
y.name	character string, for labelling the vertical axis.
debug	boolean, set to TRUE to monitor the work.
...	extra things handed down to plotting functions.

Author(s)

Dan Kelley

References

https://en.wikipedia.org/wiki/Burndown_chart

See Also

Other things related to burndown data: as.burndown(), burndown-class, burndown, read.burndown(), summary,burndown-method

Examples

```
library(plan)
data(burndown)
summary(burndown)
plot(burndown)
```

---

plot,gantt-method *Draw a Gantt diagram*

Description

Plot a gantt chart that shows the time allocated to a set of tasks, optionally also with an indication of discrete events that occur as instants in time.

Usage

```
## S4 method for signature 'gantt'
plot(
  x,
  xlim,
  time.format = NULL,
  time.labels.by,
  time.lines.by,
  event.time = NULL,
  event.label = NULL,
  event.side = 3,
  col.connector = "black",
  col.done = gray(0.3),
  col.notdone = gray(0.9),
  col.eventLine = gray(0.1),
  col.event = par("fg"),
  cex.event = par("cex"),
  font.event = par("font"),
  lty.eventLine = par("lty"),
  lwd.eventLine = par("lwd"),
  bg = par("bg"),
  grid.col = "lightgray",
  grid.lty = "dotted",
```

```
  ylabels = list(col = 1, cex = 1, font = 1, justification = 1),
  arrows = NULL,
  main = "",
  line.main = NA,
  cex.main = par("cex"),
  mgp = c(2, 0.7, 0),
  maiAdd = rep(0, 4),
  axes = TRUE,
  debug = FALSE,
  ...
)
```

Arguments

x	A gantt object.
xlim	optional range of time axis; if not provided, the range of times in x will be used.
time.format	format for dates on time axis; defaults to 3-letter month.
time.labels.by	suggested label increment on time axis, e.g. time.labels.by="2 months" to get a two-month interval. If not supplied, the axis will be generated automatically.
time.lines.by	suggested interval between vertical grid lines on the plot, e.g. time.lines.by="1 week" for weekly. If not supplied, the grid will be generated automatically.
event.time	vector of event times, e.g. conferences, whose time cannot be altered.
event.label	vector of character strings holding event names.
event.side	side for event labels.
col.connector	colour of (optional) connectors between items.
col.done	colour of work that has been done already. This may be a vector of colours, one for each item in the gantt table.
col.notdone	colour of work that has not been done yet. This may be a vector of colours, one for each item in the gantt table.
col.eventLine	colour of event lines; may be a vector.
col.event	colour of event labels; may be a vector.
cex.event	expansion factor for event labels; may be a vector.
font.event	font for event labels; may be a vector.
lty.eventLine	line type for event lines; may be a vector.
lwd.eventLine	line width for event lines; may be a vector.
bg	background colour for plot.
grid.col	colour for grid.
grid.lty	line type for grid.
ylabels	A list with elements col for colour, cex for character-expansion factor, font for font, and justification for the placement in the margin (0 means left-justified, and 1 means right-justified. (NOTE: left-justification works poorly in

RStudio, but properly in other systems.) It usually makes sense for the elements in `ylabels` to be vectors of the same length as the topic list. However, shorter vectors are permitted, and they lengthened by copying the default values at the end (see Example 6).

`arrows` A vector of strings, one for each topic, indicating the nature of the arrows that may be drawn at the ends of task bars. The individual values may be `"left"`, `"right"`, `"both"` or `"neither"`. Set `arrows=NULL`, the default, to avoid such arrows.

`main` character string to be used as chart title.

`line.main` line where title occurs. If `NA`, then the title is placed in a default location; otherwise, it is `line.main` lines above the top of the plot.

`cex.main` numeric, font-size factor for title.

`mgp` setting for par(`mgp`), within-axis spacing. The default value tightens axis spacing.

`maiAdd` inches to add to the auto-computed margins at the bottom, left, top, and right margins. The values may be negative (to tighten margins) but the sum will be truncated to remain positive.

`axes` logical, `TRUE` to draw the x axis. (Setting to `FALSE` permits detailed axis tweaking.)

`debug` logical value, `TRUE` to monitor the work.

`...` extra things handed down.

Details

Time is indicated along the x axis, and tasks are stacked along the y axis, akin to progress bars. Colour-coding can be used to indicate the degree of completion of each task. These codes can be set individually for individual tasks. Progress bars can have arrows (on either end), suggesting tasks with flexible start/end dates or overdue tasks. Vertical lines may be drawn for discreet events. See "Examples" for a few of the possibilities.

Author(s)

Dan Kelley

References

Gantt diagrams are described on wikipedia `https://en.wikipedia.org/wiki/Gantt_Chart`.

See Also

Other things related to gantt data: `as.gantt()`, `gantt-class`, `ganttAddTask()`, `gantt`, `read.gantt()`, `summary,gantt-method`

Examples

```
library(plan)
data(gantt)
summary(gantt)

# 1. Simple plot
plot(gantt)

# 2. Plot with two events
event.label <- c("Proposal", "AGU")
event.time <- c("2008-01-28", "2008-12-10")
plot(gantt, event.label=event.label,event.time=event.time)

# 3. Control x axis (months, say)
plot(gantt,labels=paste("M",1:6,sep=""))

# 4. Control task colours
plot(gantt,
     col.done=c("black", "red", rep("black", 10)),
     col.notdone=c("lightgray", "pink", rep("lightgray", 10)))

# 5. Control event colours (garish, to illustrate)
plot(gantt, event.time=event.time, event.label=event.label,
     lwd.eventLine=1:2, lty.eventLine=1:2,
     col.eventLine=c("pink", "lightblue"),
     col.event=c("red", "blue"), font.event=1:2, cex.event=1:2)

# 6. Top task is in bold font and red colour
plot(gantt,ylabels=list(col="red",font=2))

# 7. Demonstrate zero-time item (which becomes a heading)
gantt[["description"]][1] <- "Preliminaries"
gantt[["end"]][1] <- gantt[["start"]][1]
plot(gantt, ylabel=list(font=2, justification=0))

# 8. Arrows at task ends
plot(gantt, arrows=c("right","left","left","right"))
```

`read.burndown` *Scan burndown data file*

Description

Read a data file containing burndown information.

Usage

```
read.burndown(file, debug = FALSE)
```

Arguments

file	a connection or a character string giving the name of the file to load.
debug	boolean, set to TRUE to print debugging information.

Details

Reads a burndown dataset.

A strict format is required, in which the following items must be present, in the stated order, and with nothing else in the file. An example is given after the description.

- Line 1: contains two comma-separated items: the string Start, and a time expressed in ISO 8601 format (YYYY-MM-DD or YYY-MM-DD hh:mm:ss). This line indicates the start of the project.
- Line 2: as Line 1, but the string is to be Start, and the line indicates the deadline for the project.
- Line 3: a header line for a "tasks" list, comprising the following three words separated by commas: Key, Description, and Effort.
- Lines 4 to N: data lines, each containing three items: a numeric index "Key" for the task, a short "Description" of the task, and the estimated "Effort" for this task, expressed as a number. The keys must be distinct, and they must match the keys in the progress table (see below). The description should be short enough to give a reasonable-size legend as created by plot,burndown-method(). The effort may be expressed in any convenient unit, e.g. the number of hours or days for the task, or as a percentage of the overall task.
- Line N+1: a header line for the "Progress" list, comprising the following four words separated by commas: Key, Done, and Time.
- Line N+2 to end: data lines holding Progress items. Each "Key" must match a key in the task list. The "Done" column holds the percentage of the task that has been completed. The "Time" is in ISO 8601 format, as described above.

Value

A burndown object.

Sample data file

```
Start, 2006-04-08 12:00:00
Deadline, 2006-04-11 20:00:00
Key, Description,            Effort
  1, Code read.burndown(),    4
  2, Code summary.burndown(), 1
  3, Code plot.burndown(),    5
  4, Create R package,        2
  5, Write documentation,     2
  6, Set up website,          1
Key, Done, Time
  1,    5, 2006-04-08 13:00:00
  2,    5, 2006-04-08 13:30:00
```

```
1,   10, 2006-04-08 14:00:00
2,   50, 2006-04-08 15:00:00
4,    5, 2006-04-08 19:30:00
5,    5, 2006-04-08 20:00:00
4,  100, 2006-04-08 21:16:00
1,   50, 2006-04-09 09:10:00
3,    5, 2006-04-09 09:41:00
3,   30, 2006-04-09 10:18:00
3,   80, 2006-04-09 11:00:00
2,   60, 2006-04-09 12:00:00
2,  100, 2006-04-09 12:10:00
1,   70, 2006-04-09 12:30:00
5,   30, 2006-04-09 13:50:00
5,   90, 2006-04-09 14:20:00
5,  100, 2006-04-09 14:30:00
1,  100, 2006-04-09 14:35:00
3,  100, 2006-04-09 14:40:00
6,  100, 2006-04-09 16:00:00
```

Author(s)

Dan Kelley

See Also

Other things related to burndown data: `as.burndown()`, `burndown-class`, `burndown`, `plot,burndown-method`, `summary,burndown-method`

Examples

```
library(plan)
filename <- system.file("extdata", "burndown.dat", package="plan")
b <- read.burndown(filename)
summary(b)
plot(b)
```

---

read.gantt *Read a gantt data file*

Description

Read a data file containing gantt information. The data format is strict, and deviations from it may lead to error messages that are difficult to understand; see "Details".

Usage

```
read.gantt(file, debug = FALSE)
```

Arguments

file	a connection or a character string giving the name of the file to load.
debug	boolean, set to TRUE to print debugging information.

Details

The first line is a header, and must contain the words Key, Description, Start, End, Done, and NeededBy, written exactly in this way, with commas separating the words. (Blanks are ignored in this line.)

Additional lines indicate the details of each of several sub-projects, in comma-separated items, as follows:

- A key for the task. These must be distinct, and are typically just the numbers 1, 2, 3, etc.
- A description of the task. (This may not contain commas!)
- The start time for the task, in ISO 8601 format (YYYY-MM-DD or YYYY-MM-DD hh:mm:ss).
- The end time for the task, in the same format as the starting time. If an end time equals the corresponding start time, no rectangle will be drawn for the activity, and this gives a way to make headings (see example 7 for plot,gantt-method()).
- A number indicating the percentage of this task that has been completed to date.
- A space-separated optional list of numbers that indicate the keys of other tasks that depend on this one. This list is ignored in the present version of read.gantt().

Value

A gantt object, which is a data frame containing description (a character description of the task), "start" (the task's start time), "end" (the task's end time), "progress" (a number giving the percent progress on this item, or NA if none given), and needed.by (a number giving the indices of other tasks that rely on this task, or NA if none given).

Sample data file

```
Key, Description,                Start,        End, Done, NeededBy
  1, Assemble equipment,     2008-01-01, 2008-03-28, 90
  2, Test methods,           2008-02-28, 2008-03-28, 30
  3, Field sampling,         2008-04-01, 2008-08-14, 0
  4, Analyse field data,     2008-06-30, 2008-11-14, 0
  5, Write methods chapter,  2008-08-14, 2008-11-14, 0
  6, Write results chapter,  2008-10-14, 2009-01-15, 0
  7, Write other chapters,   2008-12-10, 2009-02-28, 0
  8, Committee reads thesis, 2009-02-28, 2009-03-14, 0
  9, Revise thesis,          2009-03-15, 2009-03-30, 0
 10, Thesis on display,      2009-04-01, 2009-04-15, 0
 11, Defend thesis,          2009-04-16, 2009-04-17, 0
 12, Finalize thesis,        2009-04-18, 2009-05-07, 0
```

Author(s)

Dan Kelley

See Also

Other things related to gantt data: as.gantt(), gantt-class, ganttAddTask(), gantt, plot,gantt-method, summary,gantt-method

Examples

```
library(plan)
filename <- system.file("extdata", "gantt.dat", package="plan")
g <- read.gantt(filename)
summary(g)
plot(g)
```

---

summary,burndown-method

Summarize a burndown object

Description

Print a summary of a burndown dataset.

Usage

```
## S4 method for signature 'burndown'
summary(object, ...)
```

Arguments

object	A burndown object.
...	ignored.

Author(s)

Dan Kelley

See Also

Other things related to burndown data: as.burndown(), burndown-class, burndown, plot,burndown-method, read.burndown()

Examples

```
library(plan)
data(burndown)
summary(burndown)
```

summary,gantt-method *Summarize a gantt object*

Description

Summarizes a gantt object.

Usage

```
## S4 method for signature 'gantt'
summary(object, ...)
```

Arguments

object	A gantt object.
...	ignored.

Details

Prints a summary of a gantt dataset.

Author(s)

Dan Kelley

References

https://en.wikipedia.org/wiki/Burndown_chart

See Also

Other things related to gantt data: as.gantt(), gantt-class, ganttAddTask(), gantt, plot,gantt-method, read.gantt()

Examples

```
library(plan)
data(gantt)
summary(gantt)
```

[[,plan-method *Extract Something From a plan Object*

Description

Extract something from a plan object, avoiding using the "slot" notation.

Usage

```
## S4 method for signature 'plan'
x[[i, j, ...]]
```

Arguments

x	A plan object.
i	The item to extract.
j	Optional additional information on the i item.
...	Optional additional information (ignored).

[[<-,plan-method *Replace Parts of a plan Object*

Description

Replace something within a plan object, avoiding using the "slot" notation.

Usage

```
## S4 replacement method for signature 'plan'
x[[i, j, ...]] <- value
```

Arguments

x	A plan object.
i	The item to replace.
j	Optional additional information on the i item.
...	Optional additional information (ignored).
value	The value to be placed into x, somewhere.

Index