Package: kwb.epanet (via r-universe)

August 31, 2024

Title R Package for Interfacing EPANET

Version 0.2.0

Description Functions enabling the reading and writing of EPANET (http://www.epa.gov/nrmrl/wswrd/dw/epanet.html) input files and reading of output files.

License MIT + file LICENSE

URL https://github.com/KWB-R/kwb.epanet

BugReports https://github.com/KWB-R/kwb.epanet/issues

Imports gtools, kwb.plot, kwb.utils, lattice, plot3D

Suggests covr, testthat

Remotes github::kwb-r/kwb.plot, github::kwb-r/kwb.utils

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.2

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

RemoteUrl https://github.com/KWB-R/kwb.epanet

RemoteRef HEAD

RemoteSha 0668410e4bcc7369d237f3efc194c8261ade6f98

Contents

`availableSections` *Available Sections*

Description

Names of sections available in EPANET input file

Usage

```
availableSections(inpfile)
```

Arguments

`inpfile`	full path to EPANET input file

Value

character vector containing the names of the sections contained in the EPANET input file (without brackets)

See Also

`readEpanetInputFile`

`calculateSpecificEnergyDemand`
Calculate Specific Energy Demand

Description

Calculate Specific Energy Demand

Usage

```
calculateSpecificEnergyDemand(
  waterDemand,
  totalEnergy,
  COLNAMES = list(Q = "Q.m3.per.hour.sum", E = "Kw.hr.per.m3.avg", Eff =
    "Average.Efficiency.avg")
)
```

Arguments

waterDemand	minimum constraint for water demand
totalEnergy	list element "energyTotal", as retrieved by runOptimisationStrategy
COLNAMES	list with elements *Q*, *E*, *Eff*, corresponding to discharge, energy demand and efficiency, respectively. Default: list(Q = "Q.m3.per.hour.sum", E = "Kw.hr.per.m3.avg", Eff = "Average.Efficiency.avg")

Value

list with elements *Q*, *E*, *Eff*, holding the column names of *totalEnergy*, corresponding to discharge, energy demand and efficiency, respectively. Default: list(Q = "Q.m3.per.hour.sum", E = "Kw.hr.per.m3.avg", Eff = "Average.Efficiency.avg")

---

calibrateModel *Calibrate Model*

Description

Calibrate Model

Usage

```
calibrateModel(
  configuration,
  pipeIDs = NULL,
  measured,
  pumpsToCalibrate = NULL,
  parameterName = "Diameter",
  parameterRange,
  showLivePlot = TRUE,
  ...
)
```

Arguments

configuration	EPANET parameterisation, e.g. as retrieved by readEpanetInputFile()
pipeIDs	regular expression or name of pipeID(s) to be used for calibration
measured	measurement data for all pumps as data.frame e.g. data.frame(pumpNames=c("pmpW1", "pmpW2"), measuredQ=c(140,190))
pumpsToCalibrate	regular expression or name of pumps to be used for calibration: e.g. "pmpW1"
parameterName	name of ONE EPANET pipe parameters to be calibrated: e.g. Diameter or Roughness
parameterRange	min/max range of possible calibration parameter values: 0-1; with parameterRange*cun = newParameterValue, e.g. parameter range = 0.5 -> 50% reduction of initial value of parameterName for all pipeIDs defined in calibrateModel()

showLivePlot	current calibration status is plotted if showLivePlot=TRUE. Default: TRUE
...	additional parameters to be passed to fitnessAdaptedModelConfiguration()

`checkReportFileForErrors` *Check Report File For Errors*

Description

Check Report File For Errors

Usage

```
checkReportFileForErrors(reportFile)
```

Arguments

reportFile	full path to report file

`createOptimisationResultsTable` *Create Optimisation Results Table*

Description

Create Optimisation Results Table

Usage

```
createOptimisationResultsTable(
  optimisationStrategies,
  averageWaterDemand,
  currentEnergyDemand,
  onlyBestSolutions = FALSE
)
```

Arguments

optimisationStrategies	average daily water demand in m3/h to be satisfied
averageWaterDemand	current specific energy demand
currentEnergyDemand	should only the best solutions be written to data.frame? Default: FALSE
onlyBestSolutions	return only best solutions? (default: FALSE)

curvesToText *Curves To Text*

Description

Curves To Text

Usage

```
curvesToText(curves)
```

Arguments

curves	data frame representing curves, with columns *ID*, *X_Value*, *Y_Value*, as returned by readEpanetInputFile in list element *CURVES*

defaultReportVariables *Default Report Variables*

Description

Default Report Variables

Usage

```
defaultReportVariables()
```

Value

list of report variable definitions as returned by reportVariable

epanetInputFileLines *Epanet Input File Lines*

Description

Epanet Input File Lines

Usage

```
epanetInputFileLines(inpdat, dbg = FALSE)
```

Arguments

inpdat	input data to be saved in EPANET's input file format
dbg	if TRUE, debug messages are shown. Default: FALSE

exampleInputFiles *Example Input Files*

Description

Example Input Files

Usage

```
exampleInputFiles()
```

Value

full path(s) to EPANET example input file(s)

extdata_file *Get Path to File in This Package*

Description

Get Path to File in This Package

Usage

```
extdata_file(..., must_exist = TRUE)
```

Arguments

...	parts of the file path to be passed to system.file
must_exist	if TRUE (the default) and the specified file does not exist, the program stops with an error message

getEpanetInstallationPath *Get Epanet Installation Path*

Description

Get Epanet Installation Path

Usage

```
getEpanetInstallationPath()
```

getLinkResults *Get Link Results*

Description

Get Link Results

Usage

```
getLinkResults(
  outdat,
  links,
  vars = c("q", "v", "hl", "wq", "sta", "set", "rr", "ff")
)
```

Arguments

outdat	output data read from EPANET output file, as returned by readEpanetOutputFile
links	names of links to be included in the returned data frame. You may use getNamesOfPipes, getNamesOfPumps, getNamesOfValves in order to get the names of available links
vars	acronyms of variables to be included in the returned data frame. "Q" = flow, "v" = velocity, "hl" = headloss, "wq" = avg. water quality, "sta" = status, "set" = setting, "rr" = reaction rate, "ff" = friction factor

getLinkTimeseriesFromOutputData *Get Link Timeseries From Output Data*

Description

Get Link Timeseries From Output Data

Usage

```
getLinkTimeseriesFromOutputData(outdat)
```

Arguments

outdat	data structure read from EPANET output file, as returned by readEpanetOutputFile.

getNamesOfCurves *Get Names Of Curves*

Description

Get Names Of Curves

Usage

```
getNamesOfCurves(inpdat, pattern = ".*")
```

Arguments

inpdat	imported EPANET file (as retrieved by readEpanetInputFile)
pattern	optional filter pattern (default: ".*")

getNamesOfJunctions *Get Names Of Junctions*

Description

Get Names Of Junctions

Usage

```
getNamesOfJunctions(inpdat, pattern = ".*")
```

Arguments

inpdat	imported EPANET file (as retrieved by readEpanetInputFile)
pattern	optional filter pattern (default: ".*")

getNamesOfPipes *Get Names Of Pipes*

Description

Get Names Of Pipes

Usage

```
getNamesOfPipes(inpdat, pattern = ".*")
```

Arguments

inpdat	imported EPANET file (as retrieved by readEpanetInputFile)
pattern	optional filter pattern (default: ".*")

getNamesOfPumps *Get Names Of Pumps*

Description

Get Names Of Pumps

Usage

```
getNamesOfPumps(inpdat, pattern = ".*")
```

Arguments

inpdat	imported EPANET file (as retrieved by readEpanetInputFile)
pattern	optional filter pattern (default: ".*")

getNamesOfReservoirs *Get Names Of Reservoirs*

Description

Get Names Of Reservoirs

Usage

```
getNamesOfReservoirs(inpdat, pattern = ".*")
```

Arguments

inpdat	imported EPANET file (as retrieved by readEpanetInputFile)
pattern	optional filter pattern (default: ".*")

getNamesOfTanks *Get Names Of Tanks*

Description

Get Names Of Tanks

Usage

```
getNamesOfTanks(inpdat, pattern = ".*")
```

Arguments

inpdat	imported EPANET file (as retrieved by readEpanetInputFile)
pattern	optional filter pattern (default: ".*")

getNamesOfValves *Get Names Of Valves*

Description

Get Names Of Valves

Usage

```
getNamesOfValves(inpdat, pattern = ".*")
```

Arguments

inpdat	imported EPANET file (as retrieved by readEpanetInputFile)
pattern	optional filter pattern (default: ".*")

getNodeResults *Get Node Results*

Description

Get Node Results

Usage

```
getNodeResults(outdat, nodes, vars = c("d", "h", "p", "wq"))
```

Arguments

outdat	output data read from EPANET output file, as returned by readEpanetOutputFile
nodes	names of nodes to be included in the returned data frame. You may use getNamesOfJunctions, getNamesOfReservoirs, getNamesOfTanks in order to get the names of available nodes
vars	acronyms of variables to be included in the returned data frame. "d" = demand, "h" = head, "p" = pressure, "wq" = water quality

getNodeTimeseriesFromOutputData *Get Node Timeseries From Output Data*

Description

Get Node Timeseries From Output Data

Usage

```
getNodeTimeseriesFromOutputData(outdat)
```

Arguments

outdat	data structure read from EPANET output file, as returned by readEpanetOutputFile.

getNumberOfPeriods *Number of Simulation Periods*

Description

number of simulation periods, calculated from duration and hydraulic time step both of which must be given in the [TIMES] section of the EPANET configuration

Usage

```
getNumberOfPeriods(configuration)
```

Arguments

configuration	EPANET configuration, as retrieved by readEpanetInputFile

getPipeCoordinates *Get Pipe Coordinates*

Description

Get Pipe Coordinates

Usage

```
getPipeCoordinates(inpdat)
```

Arguments

inpdat	imported EPANET file (as retrieved by readEpanetInputFile)

getPumpInfo *Get Pump Info*

Description

Get Pump Info

Usage

```
getPumpInfo(inpdat)
```

Arguments

inpdat	imported EPANET file (as retrieved by readEpanetInputFile)

getPumpPerformance *Get Pump Performance*

Description

Get time series of pump performance from EPANET result using head curves and efficiency curves as contained in EPANET input file

Usage

```
getPumpPerformance(inpdata, outdata, pumpnames)
```

Arguments

inpdata	data structure read from EPANET input file, as returned by readEpanetInputFile.
outdata	data structure read from EPANET output file, as returned by readEpanetOutputFile.
pumpnames	vector with names of pumps for which the pump performance should be evaluated

Value

data frame with columns *Q* (discharge), *H* (head), *Eff* (efficiency), *specEn* (specific efficiency), *En* (energy)

See Also

plotPumpPerformance

getSection *Get Section*

Description

Get section from EPANET input file

Usage

```
getSection(inpfile, sectionName)
```

Arguments

inpfile	full path to EPANET input file
sectionName	name of section to be read, for possible section names see the documentation of the EPANET Toolkit

Value

data frame representing the content of the section in the input file. If possible, column names are read from the section's header line

outputFileSize *Size of Binary Output File*

Description

Size of binary output file in Bytes, kB (rounded), MB (rounded)

Usage

```
outputFileSize(configuration)
```

Arguments

configuration	EPANET configuration, representing an EPANET input file, as returned by readEpanetInputFile

Value

named vector of numeric representing output file size in bytes, kB (rounded) and MB (rounded), respectively

plotCalibration *Plot Calibration*

Description

Plot Calibration

Usage

```
plotCalibration(newRes)
```

Arguments

newRes	expects data.frame object "newRes" as input parameter (is automatically produced by calibrateModel(). Required columns: *Qerror*, *calibrationRunNumber*, *pch*, *colors*, *pumpNames*

plotCurves *Plot Curves*

Description

Plot Curves

Usage

```
plotCurves(curves, curveNames = unique(curves$ID), ...)
```

Arguments

curves	data frame containing pump curves, as returned by readEpanetInputFile in list element *CURVES*
curveNames	names of curves to be plotted. Default: all names in column *ID* of *curves*
...	additional arguments passed to xyplot

plotModel *Plot Model*

Description

plot EPANET model network. Allows plotting of changed pipe ids, which are defined as vector in parameter "changedPipeIDs"

Usage

```
plotModel(
  inpdat,
  pch = 16,
  cex = 0.2,
  xlab = "",
  ylab = "",
  main = defaultMain(inpdat, 1),
  zoomToPumps = FALSE,
  changedPipeIDs = NULL,
  ...
)
```

Arguments

inpdat	EPANET input file
pch	Either an integer specifying a symbol or a single character to be used as the default in plotting points. See points for possible values and their interpretation (default: 16)
cex	A numerical value giving the amount by which plotting text and symbols should be magnified relative to the default. This starts as 1 when a device is opened, and is reset when the layout is changed, e.g. by setting mfrow. (default: 0.2)
xlab	a title for the x axis (default: "")
ylab	a title for the y axis (default: "")
main	an overall title for the plot (default: kwb.epanet:::defaultMain(inpdat, 1))
zoomToPumps	should plot zoomed to pumps (default: FALSE)
changedPipeIDs	optional, vector with changed pipe IDs (default: NULL)
...	additional arguments passed to plot

plotOptimisationResults *Plot Optimisation Results*

Description

Plot Optimisation Results

Usage

```
plotOptimisationResults(
  totalEnergy,
  name = "",
  pumpsToReplace = "",
  userConstraints,
  currentOperation,
  ...
)
```

Arguments

`totalEnergy`	list element *energyTotal* as retrieved by `runOptimisationStrategy`
`name`	name of optimisation scenario (default: "")
`pumpsToReplace`	optional vector with pump_ids to be replaced (default: "")
`userConstraints`	list of userConstraints with elements *namesOfWellsWithQualityProblems*, *waterDemand*
`currentOperation`	list of currentOperation with elements *SpecificEnergy*, (specific energy demand: kwh/m3) and *Label* ("Specific energy demand of current operation")
`...`	additional arguments passed to scatter2D

plotPumpPerformance *Plot Pump Performance*

Description

Plot time series of discharge, head and pump performance

Usage

```
plotPumpPerformance(xCols, yCols, pumpPerformanceTimeSeries)
```

Arguments

xCols	vector of columns contained in data.frame retrieved by getPumpPerformance() to be used as for x axis plotting, e.g. c("step", "Q")
yCols	vector of columns contained in data.frame retrieved by getPumpPerformance() to be used as for y axis plotting, e.g. c("Eff", "specEn", "En")
pumpPerformanceTimeSeries	pump performance time series as retrieved by getPumpPerformance()

See Also

getPumpPerformance

---

readEpanetInputFile *Read EPANET Input File*

Description

Read EPANET Input File

Usage

```
readEpanetInputFile(inpfile, dbg = FALSE)
```

Arguments

inpfile	full path to EPANET input file
dbg	if TRUE, debug messages are shown. Default: FALSE

Value

list with elements representing the different sections of the EPANET input file. The names of the list elements correspond to the names of the sections that were found in the input file. Each list element is a data frame containing the content of the corresponding section.

See Also

availableSections, readEpanetOutputFile

`readEpanetOutputFile` *Read EPANET Output File*

Description

Read EPANET Output File

Usage

```
readEpanetOutputFile(
  outfile,
  read.prolog = TRUE,
  read.energyUse = TRUE,
  read.dynamicResults = TRUE,
  read.epilog = TRUE
)
```

Arguments

`outfile`	full path to EPANET output file
`read.prolog`	if TRUE, the "Prolog" section is read from the output file and contained in the output list
`read.energyUse`	if TRUE, the "Energy Use" section is read from the output file and contained in the output list
`read.dynamicResults`	if TRUE, the "Extended Period" section is read from the output file and contained in the output list
`read.epilog`	if TRUE, the "Epilog" section is read from the output file and contained in the output list

Value

list with elements *prolog* (if `read.prolog` = TRUE), *energyUse* (if `read.energyUse` = TRUE), *dynamicResults* (if `read.dynamicResults` = TRUE) and *epilog* (if `read.epilog` = TRUE), containing the different parts of the output file, as described in the documentation of the EPANET Toolkit.

See Also

`readEpanetInputFile`

readResultsFromReportFile *Read Results From Report File*

Description

reads an EPANET report file and returns a list with node and link data

Usage

```
readResultsFromReportFile(reportFile, warn = TRUE)
```

Arguments

reportFile	full path to report file, generated by EPANET
warn	if TRUE, a warning is given if no Node or Link results were found in the report file.

Value

list with elements *nodeData* containing a data frame with node results and *linkData* containing a data frame with link results

replaceCurves *Replace Curves In CURVES Data*

Description

Replace curves in CURVES data

Usage

```
replaceCurves(curves, newCurves)
```

Arguments

curves	data frame representing curves, with columns *ID*, *X_Value*, *Y_Value*, as returned by readEpanetInputFile in list element *CURVES*
newCurves	new curves data frame representing curves, with columns *ID*, *X_Value*, *Y_Value*,

replaceCurveSectionInInputFile *Replace Curve Section In Input File*

Description

Replace Curve Section In Input File

Usage

```
replaceCurveSectionInInputFile(inpfile, newCurves)
```

Arguments

inpfile	full path to EPANET input file
newCurves	modified CURVES section. Data frame as returned by readEpanetInputFile in list element *curves*

Value

vector of character(s) representing the rows of the input file

replaceOneCurve *Replace One Curve In CURVES Data*

Description

Replace One Curve In CURVES Data

Usage

```
replaceOneCurve(curves, curveName, x, y)
```

Arguments

curves	data frame representing curves, with columns *ID*, *X_Value*, *Y_Value*, as returned by readEpanetInputFile in list element *CURVES*
curveName	name of the curve to be replaced
x	vector of x values of the curve
y	vector of y values of the curve

Value

data frame in which the lines corresponding to the curve named *curveName* are replaced with x and y values given in *x* and *y*, respectively

reportEnergyUse *Report Energy Use*

Description

Report Energy Use

Usage

```
reportEnergyUse(outdat)
```

Arguments

outdat	output data read from EPANET out-file

reportVariable *Report Variable Definition*

Description

Report variable definition for argument *variables* of `setReportOptions`

Usage

```
reportVariable(
  name = "Elevation",
  yes = TRUE,
  below = NA,
  above = NA,
  precision = 2
)
```

Arguments

name	name of variable. Node variables that can be reported on include: "Elevation", "Demand", "Head", "Pressure", "Quality". Link variables include: "Length", "Diameter", "Flow", "Velocity", "Headloss", "LinkQuality", "LinkStatus", "Setting" (Roughness for pipes, speed for pumps,, pressure/flow setting for valves), "Reaction" (reaction rate), "FFactor" (friction factor)
yes	shall the variable be importet on?
below	if set to a non-NA value (default: NA) only values below the given value will be reported on
above	if set to a non-NA value (default: NA) only values above the given value will be reported on
precision	the variable will be reported on with the given precision (number of decimal places). Default: 2

`runEpanet` *Run EPANET With Given Input File*

Description

Run EPANET With Given Input File

Usage

```
runEpanet(
  inpfile,
  returnOutput = FALSE,
  epanet.dir = getEpanetInstallationPath(),
  intern = FALSE,
  write.output = TRUE,
  ...,
  dbg = FALSE
)
```

Arguments

`inpfile`	full path to EPANET input file
`returnOutput`	if TRUE, the output is read from the generated output file and returned
`epanet.dir`	path to EPANET installation directory. Default: getEpanetInstallationPath()
`intern`	a logical, indicates whether to make the output of the command an R object.
`write.output`	if TRUE, EPANET will write a binary output file, else not
`...`	further arguments passed to `readEpanetOutputFile`, such as: *read.prolog*, *read.energyUse*, *read.dynamicResults*, *read.epilog*, see there.
`dbg`	if TRUE, debug messages are shown. Default: FALSE

`runEpanetConfiguration`
Run EPANET INP configuration

Description

Run EPANET INP configuration

Usage

```
runEpanetConfiguration(
  inpdat,
  name = "tmpEpanet",
  returnOutput = write.output,
  write.output = TRUE,
  ...,
  dbg = FALSE
)
```

Arguments

inpdat	input data as retrieved by readEpanetInputFile
name	name of input file to be generated in tempdir()
returnOutput	if TRUE, the output is read from the output file (if generated, see *write.output*) and returned. Default: value of *write.output*
write.output	if TRUE, EPANET will write a binary output file, else not
...	further arguments passed to runEpanet and finally to readEpanetOutputFile, such as: *read.prolog*, *read.energyUse*, *read.dynamicResults*, *read.epilog*, see there.
dbg	if TRUE, debug messages are shown. Default: FALSE

---

runEpanetGUI *Run Epanet GUI*

Description

Run Epanet GUI

Usage

```
runEpanetGUI(inpfile = "", epanet.dir = getEpanetInstallationPath())
```

Arguments

inpfile	path to EPANET input file
epanet.dir	EPANET directory (default: getEpanetInstallationPath)

`runEpanetOnCommandLine` *Run Epanet On Command Line*

Description

Run Epanet On Command Line

Usage

```
runEpanetOnCommandLine(
  inpfile,
  epanet.exe,
  intern = FALSE,
  write.output = TRUE,
  dbg = FALSE
)
```

Arguments

`inpfile`	path to EPANET input file
`epanet.exe`	path to EPANET executable
`intern`	a logical, indicates whether to make the output of the command an R object. (default: FALSE)
`write.output`	if TRUE, EPANET will write a binary output file, else not
`dbg`	if TRUE, debug messages are shown. Default: FALSE

`runOptimisationStrategy` *Run Optimisation Strategy*

Description

Run Optimisation Strategy

Usage

```
runOptimisationStrategy(
  configuration,
  newCurvesData,
  optimisationStrategy,
  operationSchemes = wellFieldOperationSchemes(getNamesOfPumps(configuration)),
  showLivePlot = FALSE
)
```

Arguments

configuration	EPANET configuration, representing an EPANET input file, as returned by readEpanetInputFile
newCurvesData	list with elements *Pump* (data frame with columns "ID", "X_VALUE" and "Y_VALUE") of pump curves, *GlobalPumpEfficiency* (data frame with columns "ID", "X_VALUE" and "Y_VALUE") and sublist *PumpNamePrefix* (with elements *PumpCurves* = "TDH" and *GlobalPumpEfficiency* = "Eff")
optimisationStrategy	list with elements *name* (name of optimisation strategy), *shortName* (short name of optimisation strategy) and *pumpsToReplace* (vector with pump-ids to be replaced, if none: "")
operationSchemes	possible wellfield operation schemes. Default: wellFieldOperationSchemes(getNamesOfPumps(configu
showLivePlot	(default: FALSE)

Value

list with elements *energyTotal* and *energyPerPump*

---

setEpanetInstallationPath

Set Epanet Installation Path

Description

Set Epanet Installation Path

Usage

```
setEpanetInstallationPath(epanet.dir)
```

Arguments

epanet.dir	full path to MS Access database or ODBC database name

`setReportOptions` *Set REPORT Option in EPANET configuration*

Description

Set REPORT Option in EPANET configuration

Usage

```
setReportOptions(
  configuration,
  pagesize = 0,
  file = "",
  status = "NO",
  summary = "YES",
  messages = "YES",
  energy = "NO",
  nodes = "NONE",
  links = "NONE",
  variables = defaultReportVariables()
)
```

Arguments

`configuration`	EPANET `configuration`, representing an EPANET input `file`, as returned by `readEpanetInputFile`
`pagesize`	`pagesize` sets the number of lines written per page of the output report. The default is 0, meaning that no line limit per page is in effect.
`file`	`file` supplies the name of a `file` to which the output report will be written. If the `file` name contains spaces then it must be surrounded by double quotes. If not supplied then the Report `file`, as specified in the second parameter of the ENopen (or ENepanet) function will be used.
`status`	`status` determines whether hydraulic `status` `messages` are written to the Report `file`. If YES is selected the `messages` will identify those network components that change `status` during each time step of the simulation. If FULL is selected, then convergence information will also be included from each trial of each hydraulic analysis. This level of detail is only useful for de-bugging networks that become hydraulically unbalanced. The default is NO.
`summary`	`summary` determines whether a `summary` table of number of network components and key analysis options is generated. The default is YES.
`messages`	`messages` determines whether error and warning `messages` generated during a hydraulic/water quality analysis are written to the Report `file`. The default is YES.
`energy`	`energy` determines if a table reporting average `energy` usage and cost for each pump is provided. The default is NO.

nodes	nodes identifies which nodes will be reported on. You can either list individual node ID labels or use the keywords NONE or ALL. Additional NODES lines can be used to continue the list. The default is NONE.
links	links identifies which links will be reported on. You can either list individual link ID labels or use the keywords NONE or ALL. Additional LINKS lines can be used to continue the list. The default is NONE.
variables	list of report variables as defined by reportVariable, e.g. list(reportVariable(name = "VELOCITY", above = 3.0, precision = 4), reportVariable(name = "F-FACTOR", precision = 4)). Default: defaultReportVariables

---

`setTimeParameter` *Set EPANET' Time-Related Simulation Parameters*

Description

Set EPANET' time-related simulation parameters. See EPANET reference for the meaning of the parameters

Usage

```
setTimeParameter(
  configuration,
  duration = "",
  hydraulic.timestep = "",
  quality.timestep = "",
  rule.timestep = "",
  pattern.timestep = "",
  pattern.start = "",
  report.timestep = "",
  report.start = "",
  start.clocktime = "",
  statistic = ""
)
```

Arguments

configuration	EPANET configuration, representing an EPANET input file, as returned by readEpanetInputFile
duration	(default: "")
hydraulic.timestep	(default: "")
quality.timestep	(default: "")
rule.timestep	(default: "")
pattern.timestep	(default: "")

pattern.start (default: "")
report.timestep (default: "")
report.start (default: "")
start.clocktime (default: "")
statistic (default: "")

Value

return configuration with modified [TIMES] parameterisation

---

setWellFieldOperation *Set Well Field Operation*

Description

Set Well Field Operation

Usage

```
setWellFieldOperation(config, operationScheme)
```

Arguments

config	EPANET configuration, representing an EPANET input file, as returned by readEpanetInputFile
operationScheme	one possible wellfield operation scheme (e.g. wellFieldOperationSchemes(getNamesOfPumps(configur

Value

config with modified well field operation rules

---

showProperties *Show Properties*

Description

Show node and link properties available in EPANET output

Usage

```
showProperties(outdata)
```

Arguments

outdata	list structure with EPANET results as retrieved by readEpanetOutputFile

wellFieldOperationSchemes *Well Field Operation Schemes*

Description

Well Field Operation Schemes

Usage

```
wellFieldOperationSchemes(
  pumpNames,
  niceLabels = list(searchPattern = "pmp", searchReplacement = "p")
)
```

Arguments

pumpNames	Vector of pumpNames (used in EPANET input file section "PUMPS")
niceLabels	Optionally nice labeling. List with elements *searchPattern* and *searchReplacement*. Default: list(searchPattern = "pmp", searchReplacement = "p")

writeCurves *Write Curves*

Description

Write Curves

Usage

```
writeCurves(
  epanetConfig,
  pumpCurves = NULL,
  efficiencyCurves = NULL,
  drawdownCurves = NULL,
  deleteOldCurves = TRUE,
  dbg = TRUE
)
```

Arguments

epanetConfig	EPANET configuration, as retrieved by readEpanetInputFile
pumpCurves	list(data = data.frame(ID, X_Value, Y_Value), label = "")
efficiencyCurves	list(data = data.frame(ID, X_Value, Y_Value), label = "")
drawdownCurves	list(data = data.frame(ID, X_Value, Y_Value), label = "")
deleteOldCurves	should all curves in 'epanetConfig' be deleted before writing?
dbg	show debug messages? (default: TRUE)

---

writeDrawdownCurves *Write Drawdown Curves*

Description

Write Drawdown Curves

Usage

```
writeDrawdownCurves(
  epanetConfig,
  DD,
  curveNamePrefix = "dd",
  deleteOldCurves = FALSE,
  dbg = TRUE
)
```

Arguments

epanetConfig	EPANET configuration, as retrieved by readEpanetInputFile
DD	data.frame(ID, X_Value, Y_Value)
curveNamePrefix	prefix to be used in the curve name (default: "dd")
deleteOldCurves	if TRUE all curves in epantetConfig$CURVES will be deleted before adding new ones (default: FALSE)
dbg	show debug messages? (default: TRUE)

Value

Modified EPANET configuration

writeEfficiencyCurves *Write Efficiency Curves*

Description

Write Efficiency Curves

Usage

```
writeEfficiencyCurves(
  epanetConfig,
  Eff,
  curveNamePrefix = "Eff",
  deleteOldCurves = FALSE,
  dbg = TRUE
)
```

Arguments

epanetConfig	EPANET configuration, as retrieved by readEpanetInputFile
Eff	data.frame(ID, X_Value, Y_Value)
curveNamePrefix	prefix to be used in the curve name (default: "Eff")
deleteOldCurves	if TRUE all curves in epantetConfig$CURVES will be deleted before adding new ones (default: FALSE)
dbg	show debug messages? (default: TRUE)

Value

Modified EPANET configuration

writeEpanetInputFile *Write Epanet Input File*

Description

Write Epanet Input File

Usage

```
writeEpanetInputFile(inpdat, inpfile, dbg = FALSE)
```

Arguments

inpdat	input data to be saved in EPANET's input file format
inpfile	full path to input file to be created
dbg	if TRUE, debug messages are shown. Default: FALSE

---

writeInputFileWithNewCurveSection *Write Input File With New Curve Section*

Description

Write Input File With New Curve Section

Usage

```
writeInputFileWithNewCurveSection(
  inpfile,
  newCurves,
  inpfile.new = sub("(\\.[^\\.]+)$", "_new\\1", inpfile)
)
```

Arguments

inpfile	full path to EPANET input file
newCurves	modified CURVES section. Data frame as returned by readEpanetInputFile in list element *curves*
inpfile.new	full path to modified EPANET input file. (default: \<*inpfile-without-extension_new*\>.*inp*)

Value

full path to created input file

---

writePumpCurves *Write Pump Curves*

Description

Write Pump Curves

Usage

```
writePumpCurves(
  epanetConfig,
  TDH,
  curveNamePrefix = "TDH",
  deleteOldCurves = FALSE,
  dbg = TRUE
)
```

Arguments

epanetConfig	EPANET configuration, as retrieved by readEpanetInputFile
TDH	data.frame(ID, X_Value, Y_Value)
curveNamePrefix	(default: "TDH")
deleteOldCurves	if TRUE all curves in epantetConfig$CURVES will be deleted before adding new ones (default: FALSE)
dbg	show debug messages? (default: TRUE)

Value

Modified EPANET configuration

Index