Package 'r2sbml'

November 10, 2026

Type Package

Title Read 'SBML' Models and Convert Them to Simulation Code

Version 0.1.1

Description Reads models in the Systems Biology Markup Language ('SBML') (Hucka et al. (2003) <doi:10.1093/bioinformatics/btg015>) through the bundled 'libSBML' library (Bornstein et al. (2008) <doi:10.1093/bioinformatics/btn051>). Lists a model's species, parameters, compartments, reactions, rules, events and function definitions, and converts the model into ready-to-run simulation code for 'deSolve', 'mrgsolve', 'rxode2', 'MATLAB', 'Julia' or 'ubiquity'. Models are read, not modified.

URL https://sn248.github.io/r2sbml/, https://github.com/sn248/r2sbml

BugReports https://github.com/sn248/r2sbml/issues

Encoding UTF-8

Language en-GB

License GPL (>= 2)

Copyright See file inst/COPYRIGHTS

Imports Rcpp (>= 0.12.6)

Suggests deSolve, knitr, mrgsolve, rmarkdown, rxode2, testthat (>= 3.0.0), ubiquity

VignetteBuilder knitr

LinkingTo Rcpp

Biarch TRUE

SystemRequirements cmake, libxml2

Config/roxygen2/version 8.1.1

NeedsCompilation yes

Author Satyaprakash Nayak [aut, cre, cph], Sarah M. Keating [ctb] (Author of 'libSBML' and of the example code adapted in src/convertReactions.cpp and src/getMath.cpp), Frank T. Bergmann [ctb] (Author of 'libSBML' and of the example code

adapted in src/convertReactions.cpp and src/getMath.cpp),
Lucian P. Smith [ctb] (Author of 'libSBML'),
Brett G. Olivier [ctb] (Author of 'libSBML'),
Akiya Jouraku [ctb] (Author of 'libSBML'),
Michael Hucka [ctb] (Author of 'libSBML'),
Ben Bornstein [ctb] (Author of 'libSBML'),
Jeroen Ooms [ctb] (Windows build support),
California Institute of Technology [cph] (Copyright holder of 'libSBML', 2002-2020),
EMBL European Bioinformatics Institute [cph] (Copyright holder of 'libSBML', 2009-2018),
University of Heidelberg [cph] (Copyright holder of 'libSBML', 2013-2020),
University College London [cph] (Copyright holder of 'libSBML', 2020),
Japan Science and Technology Agency [cph] (Copyright holder of 'libSBML', 2002-2005),
Gilles Vollant [cph] (Author of MiniZip, bundled with 'libSBML'),
Mathias Svensson [cph] (Contributor to MiniZip, bundled with 'libSBML'),
Even Rouault [cph] (Contributor to MiniZip, bundled with 'libSBML'),
Nathan Moinvaziri [cph] (Contributor to MiniZip, bundled with 'libSBML'),
Terry Thorsen [cph] (Contributor to MiniZip, bundled with 'libSBML'),
Info-ZIP [cph] (Copyright holder of encryption code in MiniZip, bundled with 'libSBML'),
Ludwig Schwardt [cph] (Author of the zlib iostream wrappers bundled with 'libSBML'),
Kevin Ruland [cph] (Author of the zlib iostream wrappers bundled with 'libSBML'),
M. Petra Baranski [cph] (Author of 'maddy', bundled with 'libSBML'),
Tim Gromeyer [cph] (Author of 'html2md', bundled with 'libSBML')

Maintainer Satyaprakash Nayak <satyaprakash.nayak@gmail.com>

Repository CRAN

Date/Publication 2026-11-10 08:50:02 UTC

Contents

convertReactions 3
echoSBML 4
getCmtNames 4
getCmtSizes 5
getEventMath 5
getFunctionDefinition 6
getModel 6
getNumSpecies 7
getParameterTable 8
getReactionMath 8
getReactionTable 9

getRuleMath 9
getSpeciesIC 10
getSpeciesNames 11
getSpeciesTable 11
printSBML 12

Index **13**

`convertReactions` *Convert an SBML model to simulation code*

Description

Writes the model's mass balances, rules and parameters as code for the chosen simulation tool, ready to simulate.

Usage

```
convertReactions(infile, outfile, format = "R")
```

Arguments

`infile`	input file name
`outfile`	output file name
`format`	output code format, one of 'R'/'deSolve' (default), 'mrgsolve', 'nlmixr2' (or 'rxode2'), 'MATLAB', 'Julia' or 'ubiquity'. An unrecognised format is an error.

Value

NULL invisibly

Examples

```
sbml_file <- system.file("examples", "sbmlsimple.xml", package = "r2sbml")
out_file <- tempfile(fileext = ".R")
convertReactions(sbml_file, out_file, format = "R")
out_file_mrg <- tempfile(fileext = ".cpp")
convertReactions(sbml_file, out_file_mrg, format = "mrgsolve")
out_file_rx <- tempfile(fileext = ".R")
convertReactions(sbml_file, out_file_rx, format = "nlmixr2")
out_file_m <- tempfile(fileext = ".m")
convertReactions(sbml_file, out_file_m, format = "MATLAB")
out_file_jl <- tempfile(fileext = ".jl")
convertReactions(sbml_file, out_file_jl, format = "Julia")
out_file_ub <- tempfile(fileext = ".txt")
convertReactions(sbml_file, out_file_ub, format = "ubiquity")
```

echoSBML *Check an SBML file for errors*

Description

Reads the file and prints the number of diagnostics libSBML reports, then the diagnostics themselves.

Usage

```
echoSBML(filename)
```

Arguments

filename	input file name

Value

the number of diagnostics, as an integer

Examples

```
sbml_file <- system.file("examples", "sbmlsimple.xml", package = "r2sbml")
echoSBML(sbml_file)
```

getCmtNames *Names of the compartments in a model*

Description

Returns the name where one is set and falls back to the id otherwise, the same rule getSpeciesNames() uses. Previously this returned the id unconditionally while getSpeciesNames() returned the name, so the two disagreed about what a "name" was. For a model whose compartments set no name – which is all ten of the bundled examples – the result is unchanged.

Usage

```
getCmtNames(input_model)
```

Arguments

input_model	input should be an SBML Model

Value

a character vector, one entry per compartment, in model order

Examples

```
sbml_file <- system.file("examples", "sbmlsimple.xml", package = "r2sbml")
model <- getModel(sbml_file)
getCmtNames(model)
```

`getCmtSizes` *Sizes of the compartments in a model*

Description

Reports each compartment's declared size, as the file states it.

Usage

```
getCmtSizes(input_model)
```

Arguments

`input_model`	input should be an SBML Model

Value

a numeric vector, one entry per compartment, in model order

Examples

```
sbml_file <- system.file("examples", "sbmlsimple.xml", package = "r2sbml")
model <- getModel(sbml_file)
getCmtSizes(model)
```

`getEventMath` *Print the events of a model*

Description

Prints each event's trigger and assignments as infix math.

Usage

```
getEventMath(input_model)
```

Arguments

`input_model`	input should be an SBML Model

Value

0; called for the printed output

Examples

```
sbml_file <- system.file("examples", "sbmlevent.xml", package = "r2sbml")
model <- getModel(sbml_file)
getEventMath(model)
```

---

getFunctionDefinition *Print the function definitions of a model*

Description

Prints each function definition's arguments and body as infix math.

Usage

```
getFunctionDefinition(input_model)
```

Arguments

input_model	input should be an SBML Model

Value

0; called for the printed output

Examples

```
sbml_file <- system.file("examples", "sbmlfunctiondefinition.xml", package = "r2sbml")
model <- getModel(sbml_file)
getFunctionDefinition(model)
```

---

getModel *Read an SBML model*

Description

Reads an SBML file and keeps the parsed document alive for the other query functions. A file with error- or fatal-severity diagnostics is an error; lesser diagnostics are raised as warnings.

Usage

```
getModel(filename)
```

Arguments

filename	input file name, should end with .xml

Value

an external pointer to the model, to pass to the other functions

Examples

```
sbml_file <- system.file("examples", "sbmlsimple.xml", package = "r2sbml")
model <- getModel(sbml_file)
```

---

getNumSpecies *Number of species in a model*

Description

Counts the species the model declares.

Usage

```
getNumSpecies(input_model)
```

Arguments

input_model	input should be an SBML model

Value

an integer, the number of species

Examples

```
sbml_file <- system.file("examples", "sbmlsimple.xml", package = "r2sbml")
model <- getModel(sbml_file)
getNumSpecies(model)
```

getParameterTable *Table of the parameters in a model*

Description

Lists each global parameter with its declared value and units.

Usage

```
getParameterTable(input_model)
```

Arguments

input_model	input should be an SBML Model

Value

a data frame with one row per parameter and columns `Number`, `ID`, `Name`, `Value` and `Units`

Examples

```
sbml_file <- system.file("examples", "sbmlassignmentrules.xml", package = "r2sbml")
model <- getModel(sbml_file)
getParameterTable(model)
```

getReactionMath *Print the rate laws of a model*

Description

Prints each reaction's kinetic law as infix math.

Usage

```
getReactionMath(input_model)
```

Arguments

input_model	input should be an SBML Model

Value

0; called for the printed output

Examples

```
sbml_file <- system.file("examples", "sbmlsimple.xml", package = "r2sbml")
model <- getModel(sbml_file)
getReactionMath(model)
```

getReactionTable *Table of the reactions in a model*

Description

Lists each reaction with its rate law written as infix math.

Usage

```
getReactionTable(input_model)
```

Arguments

input_model	input should be an SBML Model

Value

a data frame with one row per reaction and columns `Number`, `ID`, `Name`, `Reversible` and `RateLaw`

Examples

```
sbml_file <- system.file("examples", "sbmlsimple.xml", package = "r2sbml")
model <- getModel(sbml_file)
getReactionTable(model)
```

getRuleMath *Print the rules of a model*

Description

Prints each rule as infix math; an algebraic rule is printed as `f = 0`.

Usage

```
getRuleMath(input_model)
```

Arguments

input_model	input should be an SBML Model

Value

0; called for the printed output

Examples

```
sbml_file <- system.file("examples", "sbmlassignmentrules.xml", package = "r2sbml")
model <- getModel(sbml_file)
getRuleMath(model)
```

`getSpeciesIC` *Initial values of the species in a model*

Description

A species carries `initialAmount` *or* `initialConcentration`, never both, and the unset one reads back as NaN. Asking only for the concentration therefore returned NaN for every species in an amount-based model. This returns whichever attribute the model actually sets.

Usage

```
getSpeciesIC(input_model)
```

Arguments

`input_model`	input should be an SBML Model

Details

The value is reported **as the file states it**, with no unit conversion, so a model that mixes amount-valued and concentration-valued species yields a vector that mixes units too – `getSpeciesTable()` shows which column each species used. This deliberately differs from `convertReactions()`, which must divide an amount by the compartment volume because the ODE it generates integrates concentrations.

A species that sets neither attribute – its value comes from an initial assignment or a rule – is reported as `NA`, not 0.

Value

a named numeric vector, one entry per species, named by species id

Examples

```
sbml_file <- system.file("examples", "sbmlsimple.xml", package = "r2sbml")
model <- getModel(sbml_file)
getSpeciesIC(model)
```

getSpeciesNames *Names of the species in a model*

Description

SBML separates `id`, which every species must have, from `name`, which is an optional human-readable label. Most models in the wild set only `id`, so returning `name` alone gave a vector of empty strings. This returns the `name` where one is set and falls back to the `id` otherwise, which is how SBML tools conventionally display a species. `getCmtNames()` follows the same rule. Use `getSpeciesTable()` when you need the two columns separately.

Usage

```
getSpeciesNames(input_model)
```

Arguments

`input_model`	input should be an SBML Model

Value

a character vector, one entry per species, in model order

Examples

```
sbml_file <- system.file("examples", "sbmlsimple.xml", package = "r2sbml")
model <- getModel(sbml_file)
getSpeciesNames(model)
```

getSpeciesTable *Table of the species in a model*

Description

Lists each species with its initial value, compartment and flags.

Usage

```
getSpeciesTable(input_model)
```

Arguments

`input_model`	input should be an SBML Model

Value

a data frame with one row per species and columns `Number`, `Name`, `ID`, `InitialConcentration`, `InitialAmount`, `Compartment`, `CompartmentVol`, `BoundaryCondition` and `Constant`

Examples

```
sbml_file <- system.file("examples", "sbmlsimple.xml", package = "r2sbml")
model <- getModel(sbml_file)
getSpeciesTable(model)
```

---

printSBML *Summarise an SBML file*

Description

Prints the file's SBML level and version and how many of each component its model has.

Usage

```
printSBML(filename)
```

Arguments

`filename`	filename should be the name of the xml file

Value

0 on success and 1 when the file has errors or no model; called for the printed summary

Examples

```
sbml_file <- system.file("examples", "sbmlsimple.xml", package = "r2sbml")
printSBML(sbml_file)
```

Index

convertReactions, 3

echoSBML, 4

getCmtNames, 4
getCmtSizes, 5
getEventMath, 5
getFunctionDefinition, 6
getModel, 6
getNumSpecies, 7
getParameterTable, 8
getReactionMath, 8
getReactionTable, 9
getRuleMath, 9
getSpeciesIC, 10
getSpeciesNames, 11
getSpeciesTable, 11

printSBML, 12