2.1.1.2. wc_lang.sbml package

2.1.1.2.1. Submodules

2.1.1.2.2. wc_lang.sbml.io module

Reading/writing a WC model to/from an SBML representation

Representations include * Files * Strings

Author: Arthur Goldberg <Arthur.Goldberg@mssm.edu>

Date: 2017-09-22

License: MIT

class `wc_lang.sbml.io.Reader` [source]

Bases: `object`

Read model objects from an SBML representation

> *static* `run`(*objects*, *models*, *path=None*, *get_related=True*, *title=None*, *description=None*, *keywords=None*, *version=None*, *language=None*, *creator=None*) [source]
>
> Write a list of model objects to a string or a file in an SBML format.
>
> **Parameters:**
> - **objects** (`list`) – list of objects
> - **models** (`list`) – list of model, in the order that they should appear as worksheets; all models which are not in *models* will follow in alphabetical order
> - **path** (`str` , optional) – path of SBML file to write
> - **title** (`str` , optional) – title
> - **description** (`str` , optional) – description
> - **keywords** (`str` , optional) – keywords
> - **version** (`str` , optional) – version
> - **language** (`str` , optional) – language
> - **creator** (`str` , optional) – creator

class wc_lang.sbml.io.SBMLExchange [source]

Bases: `object`

Exchange *wc_lang* model to/from a libSBML SBML representation

static read(*document*) [source]

Read a model in a libSBML *SBMLDocument* into a *wc_lang* model.

Warning: this is an unimplemented placeholder.

static write(*objects*) [source]

Write the *wc_lang* model described by *objects* to a libSBML *SBMLDocument*.

Warning: *wc_lang* and SBML semantics are not equivalent.

Algorithm:

- validate objects
- (do not add related objects, as only certain model types can be written to SBML)
- group objects by model class
- add objects to the SBML document in dependent order

Parameters: **objects** (`list`) – list of objects

Returns: an SBMLDocument containing *objects*

Return type: `SBMLDocument`

Raises: `LibSBMLError` – if the SBMLDocument cannot be created

static write_submodel(*submodel*) [source]

Create a libSBML *SBMLDocument* containing *submodel*.

To enable use of cobrapy to solve dFBA submodels, and avoid cumbersome SBML/libSBML submodels export one *wc_lang.Submodel* into one libSBML model.

Parameters: **submodel** (`Submodel`) – a submodel

Returns: an SBMLDocument containing *submodel* as a libSBML model

Return type: `SBMLDocument`

Raises: `ValueError` – if the SBMLDocument cannot be created

class wc_lang.sbml.io.Writer [source]

Bases: `object`

Write an SBML representation of a model

static **run**(*model*, *algorithms=None*, *path=None*) [source]

Write the *model*'s submodels in SBML.

Each *Submodel* in *Model model* whose algorithm is in *algorithms* is converted into a separate SBML document. If *path* is None, then the SBML is returned in string(s), otherwise it's written to file(s) which are named *path + submodel.id + suffix*.

Parameters:
- **model** (`Model`) – a *Model*
- **algorithms** (`list` , optional) – list of *SubmodelAlgorithm* attributes, defaulting to *[SubmodelAlgorithm.dfba]*
- **path** (`str` , optional) – prefix of path of SBML file(s) to write

Returns: if *path* is None, a dictionary SBML documents as strings, indexed by submodel ids, otherwise a list of SBML file(s) created

Return type: `dict` of *str*

2.1.1.2.3. wc_lang.sbml.util module

Utilities for writing a *wc_lang* model to SBML

Includes

- Exception definitions for *wc_lang.sbml*
- Higher level functions for creating SBML objects
- Utilities for wrapping libSBML calls and initializing libSBML models

Author: Arthur Goldberg <Arthur.Goldberg@mssm.edu>

Date: 2017-11-01

Copyright: 2017, Karr Lab

License: MIT

exception **wc_lang.sbml.util.Error** [source]

Bases: `Exception`

Base class for *libSBML* exceptions

exception **wc_lang.sbml.util.LibSBMLError**(*msg*) [source]

Bases: `wc_lang.sbml.util.Error`

Exception raised when libSBML returns an error

> **__str__**() [source]
>
> Provide the Exception's msg; needed for Python 2.7, although not documented

---

class **wc_lang.sbml.util.LibSBMLInterface** [source]

Bases: `object`

Methods that compactly use libSBML to create SBML objects.

The libSBML method calls provide horribly narrow interfaces, typically exchanging one value per call, which creates extremely verbose code. These methods aggregate multiple libSBML method calls to enable more compact usage.

---

wc_lang.sbml.util.add_sbml_unit(*unit_definition*, *unit_kind*, *exponent=1*, *scale=0*, *multiplier=1.0*) [source]

Add an SBML *Unit* to an existing SBML *UnitDefinition*.

Provides the SBML level 3 version 1 default values for *exponent=1*, *scale=0*, and *multiplier=1.0*. See the libSBML documentation and the SBML specs for details.

Parameters:
- **unit_definition** (`libsbml.UnitDefinition`) – a libSBML *UnitDefinition*
- **unit_kind** (`libsbml.UnitDefinition`) – the unit kind code for the SBML unit
- **exponent** (`int` , optional) – the exponent on the SBML unit
- **scale** (`int` , optional) – the scale of the SBML unit
- **multiplier** (`float` , optional) – the multiplier of the SBML unit

Returns: the new SBML Unit

Return type: `libsbml.Unit`

Raises: `LibSBMLError` – if a libSBML calls fails

---

wc_lang.sbml.util.create_sbml_doc_w_fbc() [source]

Create an SBMLDocument that uses the 'Flux Balance Constraints' extension.

Returns: the new SBML Document

Return type: `libsbml.Unit`

Raises: `LibSBMLError` – if a libSBML calls fails

wc_lang.sbml.util.create_sbml_parameter(*sbml_model*, *id*, *value*, *units*, *name=None*, *constant=True*) [source]

Add an SBML Parameter to an SBML model.

See the libSBML documentation and the SBML specs for details.

Parameters:
- **sbml_model** (`libsbml.Model`) – a libSBML Model
- **id** (`str`) – the id of the new SBML Parameter
- **value** (`obj`) – the value of the new SBML Parameter
- **units** (`str`) – the units of the new SBML Parameter
- **name** (`str` , optional) – the name of the new SBML Parameter
- **constant** (`str` , optional) – whether the new SBML Parameter is a constant

Returns: the new SBML Parameter

Return type: `libsbml.Parameter`

Raises:
- `LibSBMLError` – if a libSBML calls fails
- `ValueError` – if a *Parameter* with id *id* is already in use

wc_lang.sbml.util.get_SBML_compatibility_method(*sbml_document*) [source]

wc_lang.sbml.util.init_sbml_model(*sbml_document*) [source]

Create and initialize an SMBL model.

Parameters: **sbml_document** (`obj`) – a *libsbml* SBMLDocument

Returns: the SBML model

Return type: `libsbml.model`

Raises: `LibSBMLError` – if calling *libsbml* raises an error

wc_lang.sbml.util.str_to_xmlstr(*str*) [source]

Convert a Python string to an XML string that can be stored as a Note in an SBML Document.

Parameters: **str** (`str`) – a string

Returns: an XML string that can be stored as a Note in an SBML Document

Return type: `str`

`wc_lang.sbml.util.wrap_libsbml`(*method*, **args*, ***kwargs*) [source]

Wrap a libSBML method so that errors in return code can be easily handled.

Unfortunately, libSBML methods that do not return data usually report errors via return codes, instead of exceptions, and the generic return codes contain virtually no information. This function wraps these methods and raises useful exceptions when errors occur.

Set *returns_int* *True* to avoid raising false exceptions or warnings from methods that return integer values.

Parameters:

- **method** (`obj`) – a reference to the *libsbml* method to execute
- **args** (`list` of *obj*) – a *list* of arguments to the *libsbml* method
- **kwargs** (`dict` of *obj*) – a *dict* of options:
- **returns_int** (`bool`, optional) – whether the method returns an integer; if *returns_int* is *True*, then an exception will not be raised if the method call returns an integer
- **debug** (`bool`, optional) – whether to print debug output

Returns: if the call does not return an error, return the *libsbml* method's return value, either an object that has been created or retrieved, or an integer value, or the *libsbml* success return code, *LIBSBML_OPERATION_SUCCESS*

Return type: `obj` or *int*

Raises:

- `LibSBMLError` – if the *libsbml* call raises an exception, or returns None, or
- returns a known integer error code != *LIBSBML_OPERATION_SUCCESS*

2.1.1.2.4. Module contents