libsbml Developer's Manual

Ben Bornstein

bornstei@cds.caltech.edu
Systems Biology Markup Language Development Group
ERATO Kitano Symbiotic Systems Project
Control and Dynamical Systems, MC 107-81
California Institute of Technology, Pasadena, CA 91125, USA
http://sbml.org/

Principal Investigators: John Doyle and Hiroaki Kitano

DRAFT

Contents

- Fast Runtime

  The Gepasi generated 100 Yeast file (2Mb; 2000 reactions http://www.gepasi.org/gep3sbml.html

3 Installation

SBML Class	C Class (typedef struct)
SBase	`SBase_t`
Model	`Model_t`
UnitDefinition	`UnitDefinition_t`
Unit	`Unit_t`
Compartment	`Compartment_t`
Parameter	`Parameter_t`
Species	`Species_t`
Reaction	`Reaction_t`
SpeciesReference	`SpeciesReference_t`
KineticLaw	`KineticLaw_t`
Rule	`Rule_t`
AssignmentRule	`AssignmentRule_t`
AlgebraicRule	`AlgebraicRule_t`
CompartmentVolumeRule	`CompartmentVolumeRule_t`
ParameterRule	`ParameterRule_t`
SpeciesConcentrationRule	`SpeciesConcentrationRule_t`

SBML Enumeration	C Enumeration (typedef enum)
UnitKind	`UnitKind_t`
RuleType	`RuleType_t`

Table 1: *SBML classes and enumerations and their corresponding C class. Italicized classes are abstract, which sets their C implementation slightly apart from the others. See SemBa[illegible]*

To instantiate (create) an object use either the

```
/**
 * Prints Species to stream (good for debugging).
 */
void
myPrintSpecies (Spece0]0d_0]0dt  *                ,  FILE  *-161(e)-90(n)-90010m0.0e d
4
4
```

void UnitDefinition_

4.5 Enumerations

4.6 Abstract Classes

The SBML specification defines three classes that have no representation apart from subclasses that specialize (inherit from) them. In OOP parlance, these types are termed abstract. The abstract SBML classes are:

contains a list of rules, but a `Rule`

void SBMLDocument_printFatals (SBMLDocument_t *d, FILE *stream)

Prints all fatals encountered during the parse af this SBMLDocument to the given stream. If no fatals have occurred, that is, if `SBMLDocument_getNumFatals(d) == 0`,

```
28
29    siari = getCurrentMillis();
30    d     = readSBML(argv[1]);
31    siop  = getCurrentMillis();
32
33    m = d->model;
34
35    printf( "File: %s\n", argv[1]);
36    printf( "       model name: %s\n",  m->name );
37    printf( "  unitDefinitions: %d\n",  Model_getNumUnitDefinitions(m())) );
38
```

SBMLWriter.h defines the following functions:

SBMLWriter_t *SBMLWriter_create (void)

7 Handling of Mathematical Foralulas and MathML

libsbml can read and write MathML 2.0 (W3C, 2000
streams, as well as translate between MathML and the text-string formulas used in SBML Level 1. This section describes the MathML and mathematics handling capabilities of the library.

7.1 Reading and Writing Formulas in Text-String Foral

```
  "1 + -2e-100 / 3",
  "1 - -foo / 3",
  "2 * foo^bar + 3.1",
  "foo()",
  "foo(1)",
  "foo(1, bar)",
  "foo(1, bar, 2^-3)",
  ""
};

ASTNode_t *n;
char     *s;
int       i;

for (i = 0; i < *formulae[i]; i++)
{
  n = SBML_parseFormula( formulae[i] ); /* Convert string to AST */
  s = SBML_formulaToString(n);          /* Convert AST back to string */

  fail_unless( !strcmp(s, formulae[i]), NULL );

  ASTNode_free(n);
  safe_free(s);
}
```

A Lists

While List convenience methods (e.g., `XXX_getNumYYY()`) are provided for every class, it is

An AST *node* is a recursive structure containing a pointer to the node's value (e.g., a number

References