

libsbml API Reference Manual

Ben Bornstein

6545 Riva Drive, Suite 160, Berkeley, CA 94705, USA
borstein@indiana.edu

<http://www.sbml.org/>

Principal Investigators: John Doyle and Hiroaki Kitano

DRAFT

August 6, 2003



Contents

1 Introduction

This manual is a reference for the `libsbml` application programming interface (API). `libsbml` is

2.2 AssignmentRule.h

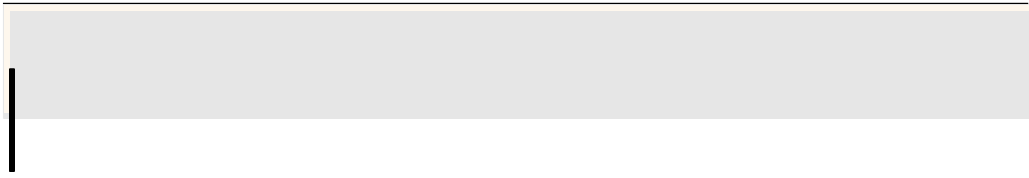


2.3 ASTNode.h

double ASTNode_getReal (const ASTNode_t *node)

Returns the value of this ASTNode as a real (double). This function should be called only when ASTNode_isReal(node) != 0.

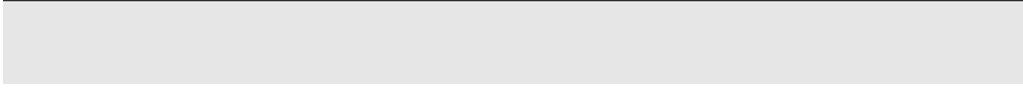
This function performs the necessary arithmetic if the node type is ASTREAL_E (mantissa 10





2.5 CompartmentVolumeRule.h

2.6 EventAssignment.h



2.7 Event.h

Event

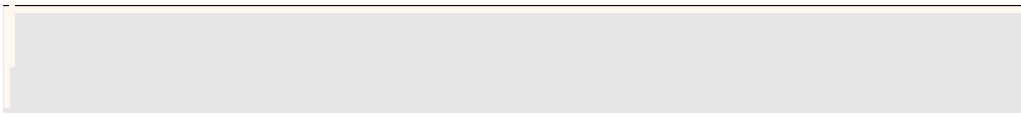
EventAssignment_t * Event_getEventAssignment (const Event_t *e, unsigned int n)

Returns the nth EventAssignment of this Event.

unsigned int Event_getNumEventAssignments (const Event_t *e)

Returns the number of EventAssignments in this Event.

2.10 FormulaTokenizer.h



2.13 List.h

List `list (void)`

Creates a new List and returns a pointer to it.

...

2.15 MathMLDocument.h

2.16 MathMLReader.h

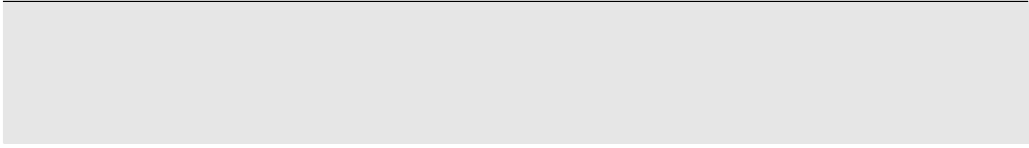
MathMLDocument_t * readMathMLFromStrTng (const char *xml)

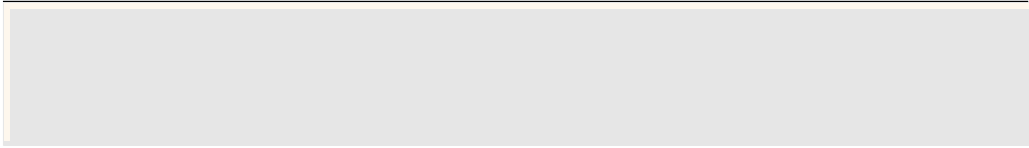
Reads the MathML from the given XML strTng, constructs a corresponding abstract syntax tree and returns a pointer to the root of the tree.

2.17 Model.h

Model_t * Model_create (void)

Creates a new Model and returns a pointer to it.





Parameter_

2.18 ModifierSpeciesReference.h

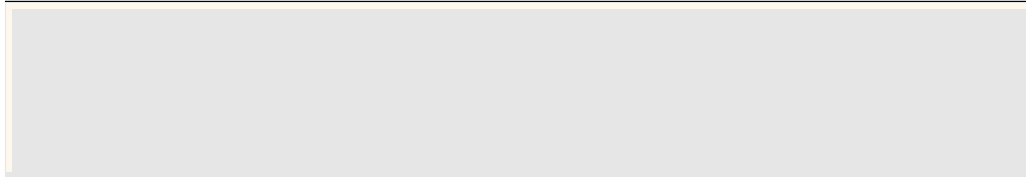


2.19 ParameterRule.h

2.20 Parameter.h

Parameter_t * Parameter_create (void)

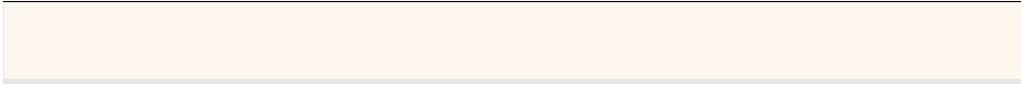
Creates a new Parameter and returns a pointer to it.



```
int Parameter_isSetValue (const Parameter_t *p)
```

Returns 1 if the value of this Parameter has been set, 0 otherwise.

2.23 Reaction.h



int Reaction_isSetKineticLaw (const Reaction_t *r)

Returns 1 if the KineticLaw of this Reaction has been set, 0 otherwise.

int Reaction_isSetFast (const Reaction_t *r)

Returns 1 if the fast status of this Reaction has been set, 0 otherwise.

In L1, fast is optional with a default of false, which means it is effectively always set.



2.25 RuleType.h

2.26 SBase.h

void SBase_init (SBase_t *sb, SBMLTypeCode_t tc)

SBase "objects" are abstract, i.e., they are not created. Rather, specific "subclasses" are created (e.g., Model) and their SBASE_FIELDS are initialized with this function. The type of the specific "subclass" is indicated by the given SBMLTypeCode.

void SBase_clear (SBase_t *sb)

Clears (frees) only the SBASE_FIELDS of sb.

SBMLTypeCode_t SBase_getTypeCode (const SBase_t *sb)

Returns the type of this SBML object.

2.27 SBMLDocument.h

SBMLDocument_t * SBMLDocument_create (void)

Creates a new SBMLDocument and returns a pointer to it.



2.28 SBMLReader.h





2.29 SBMLWriter.h



2.30 SimpleSpeciesReference.h

```
const char * SimpleSpeciesReference_getSpecies (const SimpleSpeciesReference_t *ssr)
```

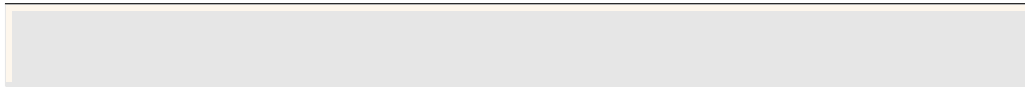
Returns the species for this SimpleSpeciesReference_t

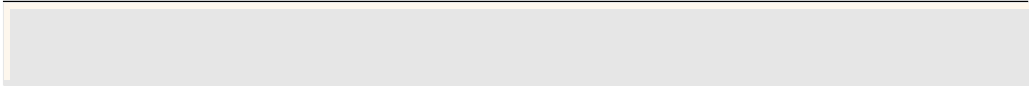
2.32 SpeciesReference.h




```
void SpeciesReference_setStoichiometry (SpeciesReference_t *sr, double value)
```

2.33 Species.h





2.34 Stack.h

Stack_t * Stack_create (int capacity)

Creates a new Stack and returns a pointer to it.

void Stack_free (Stack_t *s)

Frees the given Stack.

This function does not free individual Stack items. It frees only the Stack_t structure.


```
char * StringBuffer::getBuffer (const StringBuffer_t *sb)
```

Returns the underlying buffer contained in this `StringBuffer`.

The builder is not owned by the caller and should not be modified or deleted. The caller may take ownership of the builder by freeing the `String.Builder` directly, e.g.:

```
char buffer = StringBuffer_getBuffer(sb); safe_
```

si ser di reernedeconng

2.36 UnitDefinition.h

void UnitDefinition


```
void Unit.setScale (Unit_t *u, int value)
```

Sets the scale of this Unit to the given value.

```
void Unit.setMultiplier (Unit_t *u, double value)
```

Sets the multiplier of this Unit to the given value.

```
void Unit.setOffset (Unit_t *u, double value)
```

Sets the offset of this Unit to the given value.

