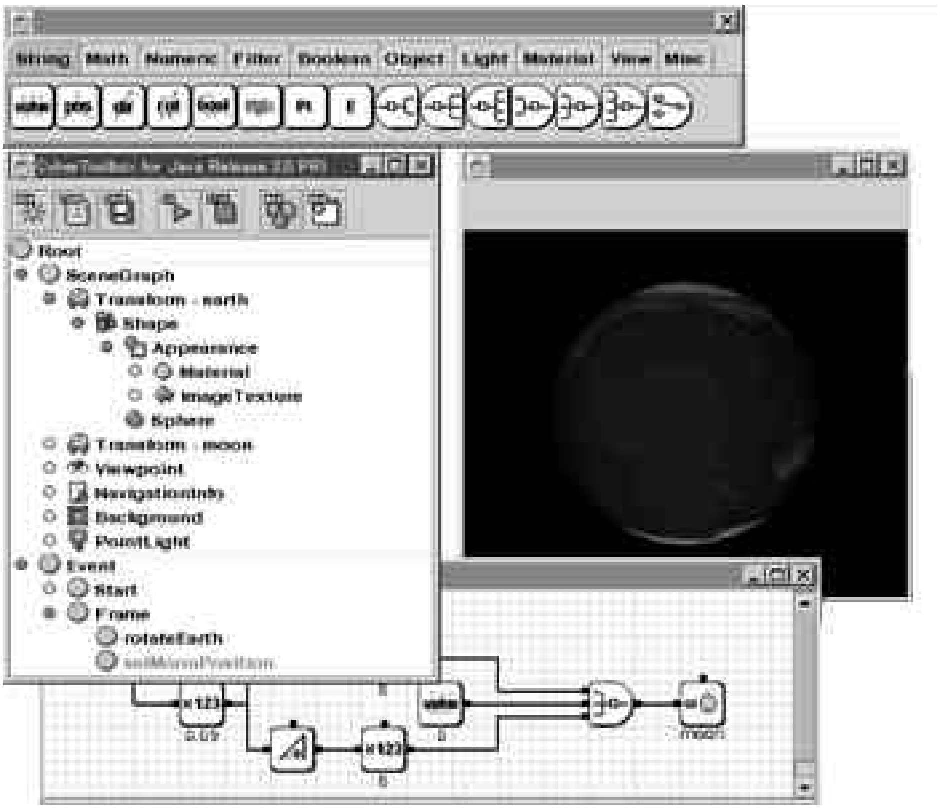


CyberToolbox

Release 2.0

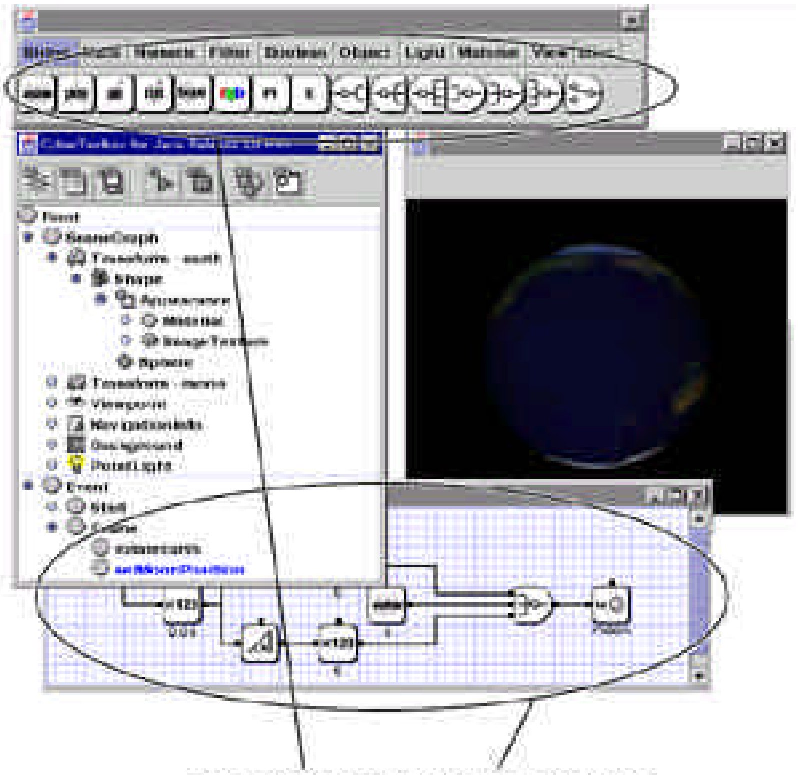
Java

User's Guide

What is CyberToolbox ?

CTB, CyberToolbox, for Java is a VRML2.0/97 authoring tool for WIN32 and Java platforms. VRML is a standard 3D file format on the internet now, and you can create some interactive behaviors, but you have to use the script programming languages, Java or Java Script, to create more good contents.

However, CTB has a icon-based programming language to solve the programming language issue, you can create the good behaviors easily. Using CTB, you can create the behaviors visually only by mouse operations.



Icon-based programming language

I am developing the CTBs with CyberVRML97 which is development libraries for C++ and Java. If you have any interest in theVRML application development, yon can get the informations in more detail from my website, http://www.cyber.koganei.tokyo.jp.

Installation

To run CTB for Java platforms, you have to install latest JDK 1.2 and Java3D packages. If you don' install the packages yet, get the packages from Sun's Java site (http://java.sun.com),..

CTB's package is distributed as a jar file. To extract the package, use a jar tool of JDK utility or WinZip program. If you want to use the jar tool to extract the package……..

```
jar xvf ctb200??.zip
```

Next, you have to add two class file packages which are included the CTB's package. "ctbvrml.jar" and "ctbmodule.jar", into your CLASSPATH setting. For example,

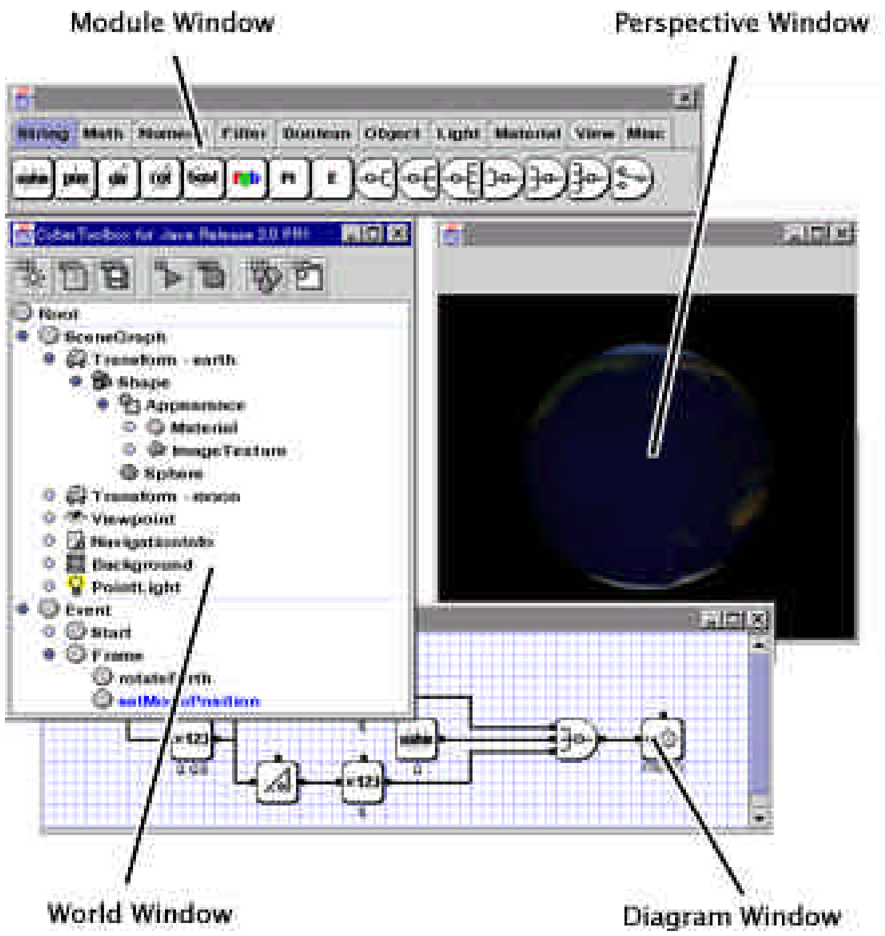
```
set CLASSPATH=.¥;ctbvrml.jar;ctbmodule.jar; .............
```

Finally, execute World class using java tool to run CTB.

```
java World
```

Operation Overview

CTB for Java platforms has four windows, World window, Perspective window, Diagram window and Module window.

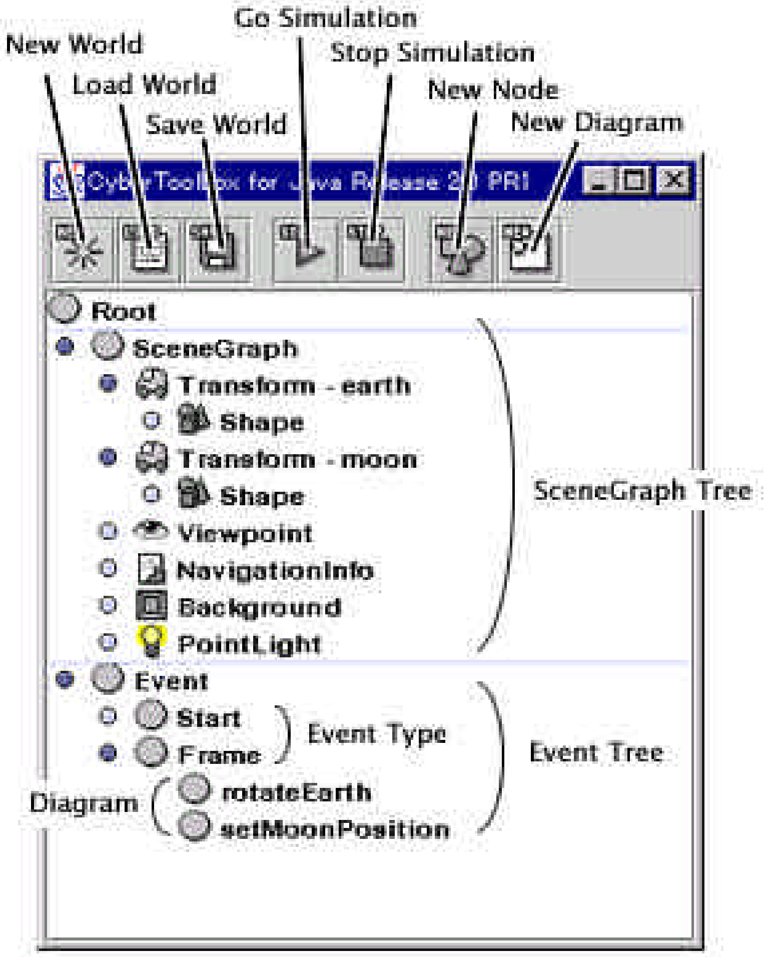


World window shows current scenegraph and behavior informations using a tree view of JFC, Perspective window shows the infomations visually using Java3D.

Diagram window is a workspace which you can create behaviors using behavior modules in Module Window. To create the behaviors, drag modules you want, drop the modules into a diagram windows, and connect between the module nodes using a mouse.

World Window

World window is a main window of CTB. The window shows current scenegraph and diagram infomations, you can add new scenegraph infomations from VRML files into a current world, save the current world into a VRML file, add new nodes, edit node infomations, start and stop the simulation, create new diagrams that are workspece to create beheviors.



To cofirm or edit node field infomations in the scenegraph tree, double-click on the node to open the setting dialog. To move a node under other parent nodes, drag the node that you want to move, and drop on the parent node of the dragging node. To confirm or edit behaviors in a diagram, double-click on the diagram to open the diagram window.

New World

Use to initialize the current world. The initialization delete all nodes, diagrams, and modules. The world became empty.

Load World

Use to load a VRML 2.0/97 file to add the all nodes into the current world.

Save World

Use to save the current world into a VRML 97 file.

Go Simulation

Use to active the current simulation to execute the world behavior actions. When the simulation is active, you can not create any new events and diagrams, edit diagrams. If you want to do the operations, you should stop the simulation.

Stop Simulation

Use to inactive the current simulation.

New Node

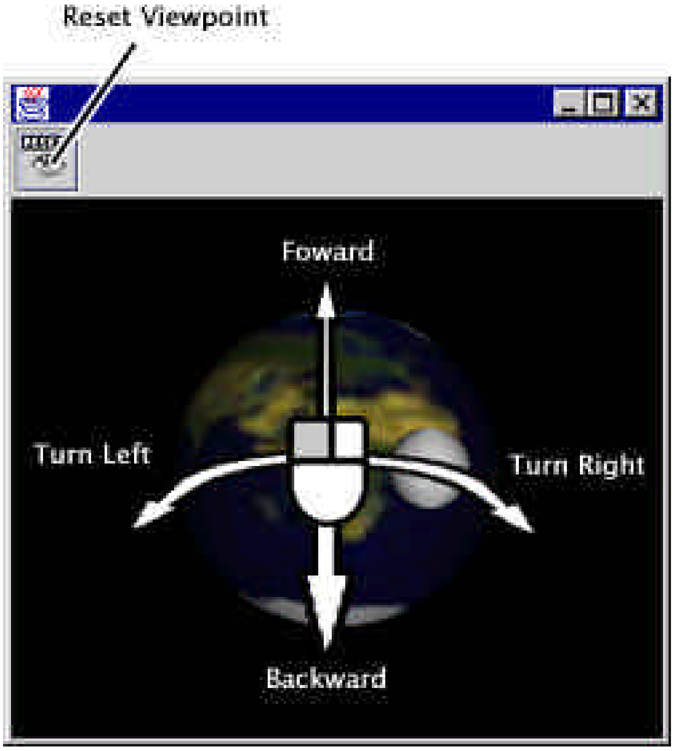
Use to add a new node as a current selected node's child. Only nodes which you can add into the patent node are shown in the dialog.

New Diagram

Use to add a new diagram, you should select the event type and set the name. If the the same diagram has been added already, you can't add.

Perspective Window

Perspective wndow shows the current virtual world using Java3D. When the simulation is active, the window shows with behaviors. Drag a moouse pointer with the left button to walk in the world.

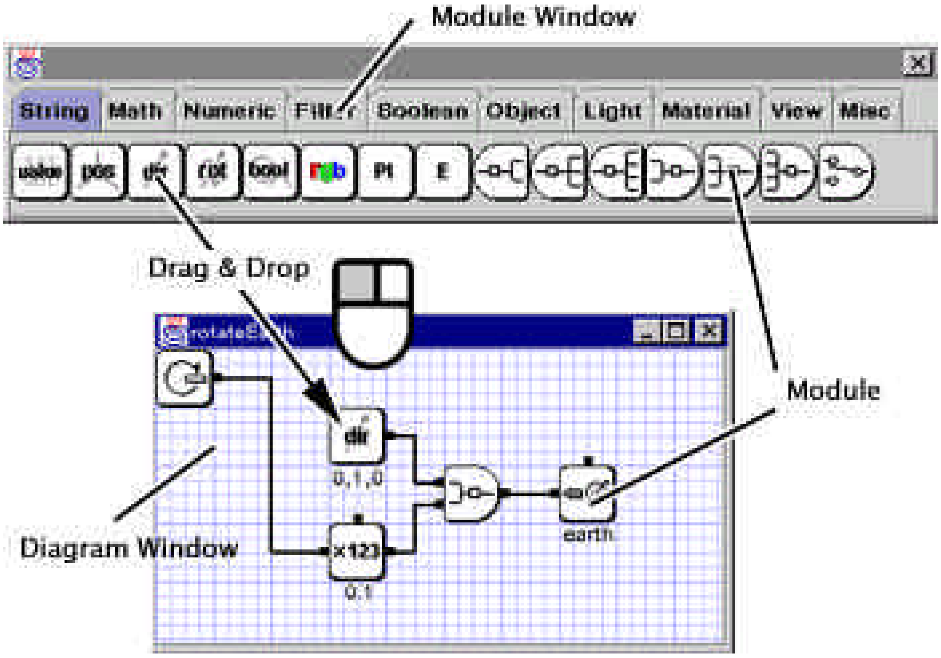


Reset Viewpoint

Use to set a current viewpoint position which you can see all geometries in the world. First, the viewpoint be moved to the bounding box center in the world. Secondly, the viewpoint be translated along +Z axis in world frame.

Diagram / Module Window

Diagram window is workspace which you can create behaviors in a current vitual world. You can create the behaviors to connect between modules of Module window



To add a new module into a diagram window, drag the module in Module window, and drop on the diagram window. To move the module position in diagram window, drag the module.

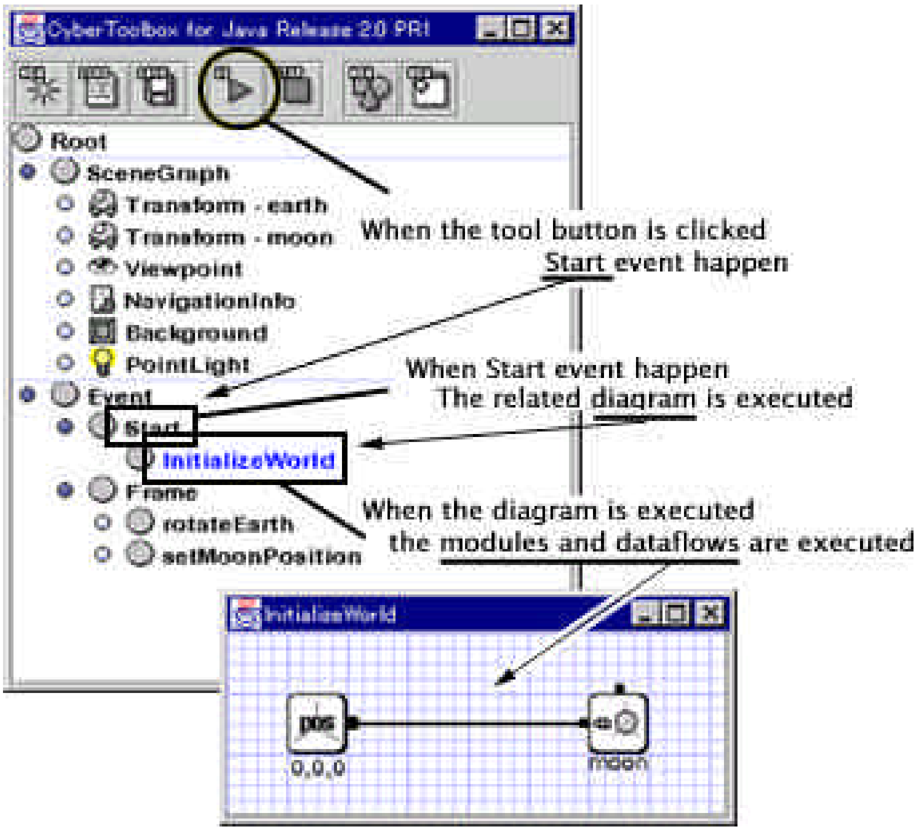
The some modules has a setting dialog to set the inside value or the target node. To open the dialog, double-click on the module.

The connected line is a data-flow line between module nodes. To connect between the nodes, drag the node, and drop on the other node.

To delete a module or a connected line, push DEL key after selecting the module or the node to click.

Behavior Overview

Using CTB, you can create fun behaviors easily. The behaviors are executed when a related event happen. When the event happen, the event execute modules and data-flows in the diagrams which are related the event.



When the tool button is clicked Start event happen

When Start event happen The related diagram is executed

When the diagram is executed the modules and dataflows are executed

Event

CTB for Java platforms has only two system events in current release, Start and Frame. In the final release, I will add more useful events which CTB for WIN32 has.

Start

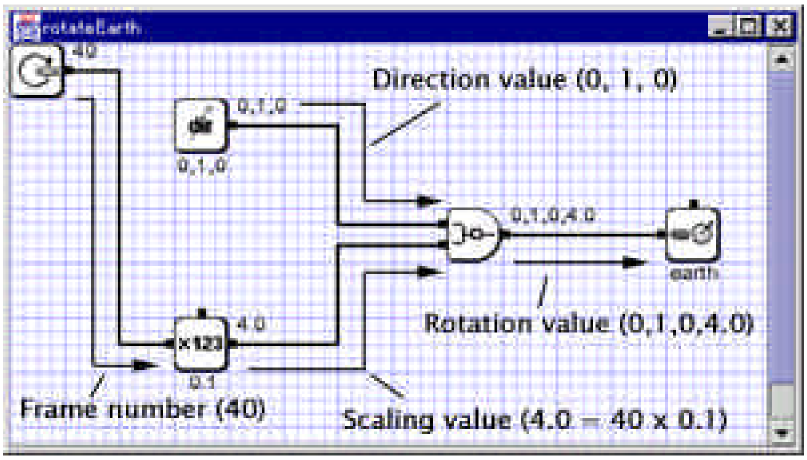
Start event happen at once when the simulation is started to click the tool button, Go Simulation, in World window. Use the event if you want to create behaviors when simulation is started at once.

Frame

Frame event happen at ten times per second after the simulation is started. The related diagram has a system module as default. The module output a current frame number.

Diagram

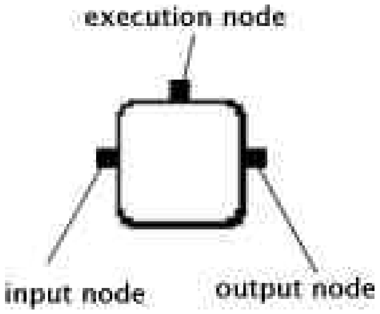
Diagram is a workspace which you can create behaviors using modules in Module window. The connected line between the module nodes is a dateflow line, the 7module send a output node data to a input node of the the other module.


rotateEarth
40
Direction value (0, 1, 0)
0,1,0
dir
0,1,0
0,1,0,4.0
earth
Rotation value (0,1,0,4.0)
×123
4.0
0.1
Frame number (40)
Scaling value (4.0 = 40 x 0.1)

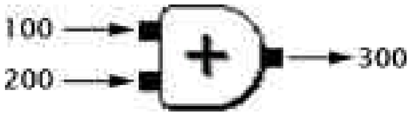
The most top module in the dataflow is executed at first, the module send the output data into other modules whick are connected the dataflow line with the module output nodes, then the other modules are executed in dataflow sequence.

Module

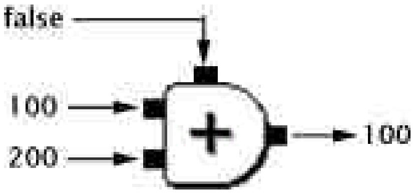
Module is a minimum unit to create behaviors, the module has three node types, a input node, a output node and a execution node.



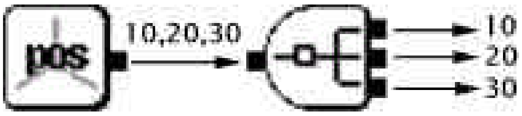
The input node type input a data from the other modules, the output node type output a data which is caluculated using the input data. For example, a following module has two input nodes and a output node. The Result is a data which is added two input values.



Using the execution node type, you can set if the module calucuation is executed. If the execution node is not connected with a dataflow line from the other module, the caluculation is executed. When the execution node is connected, the caluculation is executed when the input data is “ture”, the execution is not executed when the input data is not “true”. For example, a following module has two input nodes, a output node and a execution node which is inputed “false”. The Result is a input value data because the calclulaion is not executed..



The all node data format are strin. When a module have to caluculate the string data as a number, the module convert the string data into a number, then the module start the caluculation. The string data can has some numbers to merge the numbers into a string using canma (‘,’), you can merge some numbers into a string, or divide a string into some number strings. For example, a following left module output a position string which has three numbers, and the right module divide the string into three number string.



Module Behavior Overview

Modules are classified into nine classes, String, Numeric, Math, Filter, Boolean, Object, Material, Light, View.

The module behaviors are below. If the module has a setting dialog, you can set the value or the target node using the dialog to double-click on the module.

String

Value

Input node names	-
Output node names	OutValue
Target node	-
Result	OutValue = User setting value
Execution node	-
Setting dialog	O

Position

Input node names	-
Output node names	OutValue
Target node	-
Result	OutValue = User setting value (x, y, z)
Execution node	-
Setting dialog	O

Direction

Input node names	-
Output node names	OutValue
Target node	-
Result	OutValue = User setting value (x, y, z)
Execution node	-
Setting dialog	O

Rotation

Input node names	-
Output node names	OutValue
Target node	-
Result	OutValue = User setting value (x, y, z, angle)
Execution node	-
Setting dialog	O

Bool

Input node names	-
Output node names	OutValue
Target node	-
Result	OutValue = User setting value (true or false)
Execution node	-
Setting dialog	O

Color

Input node names	-
Output node names	OutValue
Target node	-
Result	OutValue = User setting value (r, g, b)
Execution node	-
Setting dialog	O

PI

Input node names	-
Output node names	OutValue
Target node	-
Result	OutValue = PI
Execution node	-
Setting dialog	O

E

Input node names	-
Output node names	OutValue
Target node	-
Result	OutValue = E
Execution node	-
Setting dialog	O

Divide2Values

Input node names	InValue (value1,value2)
Output node names	OutValue1 OutValue2
Target node	-

Result	OutValue1 = value1
	OutValue2 = value2

Execution node	-
Setting dialog	-

Example	InValue = 100,200
	OutValue1 = 100
	OutValue2 = 200

Divide3Values

Input node names	InValue (value1,value2, value3)
Output node names	OutValue1 OutValue2 OutValue3
Target node	-

Result	OutValue1 = value1
	OutValue2 = value2
	OutValue3 = value3

Execution node	-

Setting dialog	-

Example	InValue = 100,200,300
	OutValue1 = 100
	OutValue2 = 200
	OutValue3 = 300

Divide4Values

Input node names	InValue (value1,value2, value3,value4)
Output node names	OutValue1 OutValue2 OutValue3 OutValue4
Target node	-

Result	OutValue1 = value1
	OutValue2 = value2
	OutValue3 = value3
	OutValue4 = value4

Execution node	-
Setting dialog	-

Example	InValue = 100,200,300,400
	OutValue1 = 100
	OutValue2 = 200
	OutValue3 = 300
	OutValue4 = 400

Merge2Values

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-
Result	OutValue = InValue1,InValue2
Execution node	-
Setting dialog	-

Example	InValue1 = 100
	InValue2 = 200
	OutValue1 = 100,200

Merge3Values

Input node names	InValue1 InValue2 InValue3
Output node names	OutValue
Target node	-
Result	OutValue = InValue1,InValue2,InValue3
Execution node	-
Setting dialog	-

Example	InValue1 = 100
	InValue2 = 200
	InValue3 = 300
	OutValue1 = 100,200,300

Merge4Values

Input node names	InValue1 InValue2 InValue3 InValue4
Output node names	OutValue
Target node	-
Result	OutValue = InValue1,InValue2,InValue3,InValu4
Execution node	-
Setting dialog	-

Example	InValue1 = 100
	InValue2 = 200
	InValue3 = 300
	InValue4 = 400
	OutValue1 = 100,200,300,400

Selector

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = InValue1
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = InValue1
	    else
	        OutValue = InValue2
	}

Execution node	O
Setting dialog	-

Example	InValue1 = 100
	InValue2 = 200
	ExecutionNode = “false”
	OutValue = 200

Numeric

Add

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	   OutValue = InValue1 + InValue2
	else {
	   if (ExecutionNode data is “true”)
	      OutValue = InValue1 + InValue2
	   else
	      OutValue = InValue1
	}

Execution node	O
Setting dialog	-

Example	Example 1:
	   InValue1 = 100
	   InValue2 = 200
	   OutValue = 300
	
	Example 2:
	   InValue1 = 100,200,300
	   InValue2 = 400, 500,600
	   OutValue = 500,700,900

Minus

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	  OutValue = InValue1 - InValue2
	else {
	  if (ExecutionNode data is “true”)
	    OutValue = InValue1 - InValue2
	  else
	    OutValue = InValue1
	}

Execution node	O
Setting dialog	-

Example	Example 1:
	  InValue1 = 200
	  InValue2 = 100
	  OutValue = 100
	
	Example 2:
	  InValue1 = 600,700,800
	  InValue2 = 400, 500,600
	  OutValue = 200,200,200

Multi

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	  OutValue = InValue1 x InValue2
	else {
	  if (ExecutionNode data is “true”)
	    OutValue = InValue1 x InValue2
	  else
	    OutValue = InValue1
	}

Execution node	O
Setting dialog	-

Example	Example 1:
	InValue1 = 20
	InValue2 = 30
	OutValue = 600
	
	Example 2:
	InValue1 = 100,200,300 (pos or vector)
	InValue2 = 2
	OutValue = 200,400,600
	
	Example 3:
	InValue1 = 0,0,1 (vector)
	InValue2 = 0,1,0,1.57 (rotation)
	OutValue = 1,0,0

÷ Divide

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	OutValue = InValue1 / InValue2
	else {
	if (ExecutionNode data is “true”)
	OutValue = InValue1 / InValue2
	else
	OutValue = InValue1
	}

Execution node	O
Setting dialog	-

Example	Example 1:
	InValue1 = 600
	InValue2 = 30
	OutValue = 20
	
	Example 2:
	InValue1 = 200,400,600 (pos or vector)
	InValue2 = 2
	OutValue = 100,200,300

Mod

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	OutValue = InValue1 % InValue2
	else {
	if (ExecutionNode data is “true”)
	OutValue = InValue1 % InValue2
	else
	OutValue = InValue1
	}

Execution node	O
Setting dialog	-

Example	InValue1 = 10
	InValue2 = 3
	OutValue = 1

& And

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	  OutValue = InValue1 & InValue2
	else {
	  if (ExecutionNode data is “true”)
	    OutValue = InValue1 & InValue2
	  else
	    OutValue = InValue1
	}

Execution node	O
Setting dialog	-

Example	InValue1 = 1
	InValue2 = 2
	OutValue = 0

I Or

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	  OutValue = InValue1 | InValue2
	else {
	  if (ExecutionNode data is “true”)
	    OutValue = InValue1 | InValue2
	  else
	    OutValue = InValue1
	}

Execution node	O
Setting dialog	-

Example	InValue1 = 1
	InValue2 = 2
	OutValue = 3

Xor

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = InValue1 ^ InValue2
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = InValue1 ^ InValue2
	    else
	        OutValue = InValue1
	}

Execution node	O
Setting dialog	-

Example	InValue1 = 1
	InValue2 = 2
	OutValue = 3

Math

a++ Increment

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	   OutValue = InValue + 1
	else {
	   if (ExecutionNode data is “true”)
	      OutValue = InValue + 1
	   else
	      OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue = 1.1
	OutValue = 2.1

a-- Decrement

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	   OutValue = InValue - 1
	else {
	   if (ExecutionNode data is “true”)
	      OutValue = InValue - 1
	   else
	      OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue = 1
	OutValue = 0

|a| Abs

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = | InValue |
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = | InValue |
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue = -1
	OutValue = 1

-a Negative

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = - InValue
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = - InValue
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue = 1
	OutValue = -1

Pow

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = pow(InValue1, InValue2)
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = pow(InValue1, InValue2)
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue1 = 2
	InValue2 = 3
	OutValue = 8

Sqrt

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = sqrt(InValue)
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = sqrt(InValue)
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue = 9
	OutValue = 3

Min

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected) {
	    if (InValue1 < InValue2)
	        OutValue = InValue1
	    else
	        OutValue = InValue2
	    OutValue = sqrt(InValue)
	}
	else {
	    if (ExecutionNode data is “true”) {
	        if (InValue1 < InValue2)
	            OutValue = InValue1
	        else
	            OutValue = InValue2
	    }
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue1 = 100
	InValue2 = 200
	OutValue = 100

Max

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected) {
	   if (InValue1 > InValue2)
	      OutValue = InValue1
	   else
	      OutValue = InValue2
	   OutValue = sqrt(InValue)
	}
	else {
	   if (ExecutionNode data is “true”) {
	      if (InValue1 > InValue2)
	         OutValue = InValue1
	      else
	         OutValue = InValue2
	   }
	   else
	      OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue1 = 100
	InValue2 = 200
	OutValue = 100

Log

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	   OutValue = log(InValue)
	else {
	   if (ExecutionNode data is “true”)
	      OutValue = log(InValue)
	   else
	      OutValue = InValue
	}

Execution node	O
Setting dialog	-

Exp

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = exp(InValue)
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = exp(InValue)
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

Sin

Input node names	RadianAngle
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = sin(RadianAngle)
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = sin(RadianAngle)
	    else
	        OutValue = RadianAngle
	}

Execution node	O
Setting dialog	-

Cos

Input node names	RadianAngle
Output node names	OutValue

Target node	-

Result	if (ExecutionNode is not connected)
	   OutValue = cos(RadianAngle)
	else {
	   if (ExecutionNode data is “true”)
	      OutValue = cos(RadianAngle)
	   else
	      OutValue = RadianAngle
	}

Execution node	O
Setting dialog	-

Tan

Input node names	Radian
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	   OutValue = tan(RadianAngle)
	else {
	   if (ExecutionNode data is “true”)
	      OutValue = tan(RadianAngle)
	   else
	      OutValue = RadianAngle
	}

Execution node	O
Setting dialog	-

ASin

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = asin(InValue)
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = asin(InValue)
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

ACos

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = acos(InValue)
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = acos(InValue)
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

ATan

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	OutValue = atan(InValue)
	else {
	if (ExecutionNode data is “true”)
	OutValue = atan(InValue)
	else
	OutValue = InValue
	}

Execution node	O
Setting dialog	-

Degree2Radiun

Input node names	DegreeAngle
Output node names	OutValue
Target node	-
Result	OutValue = DegreeAngle / 180 x PI
Execution node	-
Setting dialog	-

Radiun2Degree

Input node names	DegreeAngle
Output node names	OutValue
Target node	-
Result	OutValue = DegreeAngle / 180 x PI
Execution node	-
Setting dialog	-

Filter

Scale

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = InValue * User setting value
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = InValue * User setting value
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	O

Example	InValue = 10
	User setting value = 20
	OutValue = 200

Ceil

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = ceil(InValue)
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = ceil(InValue)
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue = 12.3
	OutValue = 13

Floor

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = floor(InValue)
	else {
	    if (ExecutionNode data is “true”)
	        OutValue = floor(InValue)
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue = 12.3
	OutValue = 12

High

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected) {
	    if (User setting high value < InValue)
	        OutValue = User setting high value
	    else
	        OutValue = InValue
	}
	else {
	    if (ExecutionNode data is “true”) {
	        if (User setting hi value < InValue)
	            OutValue = User setting high value
	        else
	            OutValue = InValue
	    }
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue = 120
	User setting high value = 100
	OutValue = 100

Low

Input node names	InValue
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected) {
	    if (InValue < User setting low value)
	        OutValue = User setting low value
	    else
	        OutValue = InValue
	}
	else {
	    if (ExecutionNode data is “true”) {
	        if (InValue M ¥¥< User setting low data)
	            OutValue = User setting log value
	        else
	            OutValue = InValue
	    }
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue = 12.3
	OutValue = 12

Ragne

Target node	-

Result	if (ExecutionNode is not connected) {
	    if (InValue < User setting low value)
	        OutValue = User setting low value
	    else
	        OutValue = InValue
	    if (User setting high value < InValue)
	        OutValue = User setting high value
	    else
	        OutValue = InValue
	}
	else {
	    if (ExecutionNode data is “true”) {
	        if (InValue M ¥¥< User setting low data)
	            OutValue = User setting log value
	        else
	            OutValue = InValue
	        if (User setting high value < InValue)
	            OutValue = User setting high value
	        else
	            OutValue = InValue
	    }
	    else
	        OutValue = InValue
	}

Execution node	O
Setting dialog	-

Example	InValue = 12.3
	OutValue = 12

ScalarInterpolator

Input node names	Fraction
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = (User setting value0 –
	                    User setting value1) x Fraction
	else {
	    if (ExecutionNode data is “true”) {
	        OutValue = (User setting value0 –
	                        User setting value1) x Fraction
	    }
	    else
	        OutValue = Fraction
	}

Execution node	O
Setting dialog	O

Position2Dinterpolator

Input node names	Fraction
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	    OutValue = (User setting value0 –
	                    User setting value1) x Fraction
	else {
	    if (ExecutionNode data is “true”) {
	        OutValue = (User setting value0 –
	                        User setting value1) x Fraction
	    }
	    else
	        OutValue = Fraction
	}

Execution node	O
Setting dialog	O

Position3Dinterpolator

Input node names	Fraction
Output node names	OutValue

Target node	-

Result	if (ExecutionNode is not connected)
	OutValue = (User setting value0 –
	User setting value1) x Fraction
	else {
	if (ExecutionNode data is “true”) {
	OutValue = (User setting value0 –
	User setting value1) x Fraction
	}
	else
	OutValue = Fraction
	}

Execution node	O
Setting dialog	O

OrientationInterpolator

Input node names	Fraction
Output node names	OutValue
Target node	-

Result	if (ExecutionNode is not connected)
	OutValue = (User setting value0 –
	User setting value1) x Fraction
	else {
	if (ExecutionNode data is “true”) {
	OutValue = (User setting value0 –
	User setting value1) x Fraction
	}
	else
	OutValue = Fraction
	}

Execution node	O
Setting dialog	O

Boolean

== Equal

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (InValue1 == InValue2)
	OutValue = “true”
	else
	OutValue = “false”

Execution node	-
Setting dialog	-

!= NotEqual

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (InValue1 != InValue2)
	OutValue = “true”
	else
	OutValue = “false”

Execution node	-
Setting dialog	-

< Greater

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (InValue1 > InValue2)
	OutValue = “true”
	else
	OutValue = “false”

Execution node	-
Setting dialog	-

> Less

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (InValue1 < InValue2)
	OutValue = “true”
	else
	OutValue = “false”

Execution node	-
Setting dialog	-

≤ Equal Greater

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (InValue1 >= InValue2)
	OutValue = “true”
	else
	OutValue = “false”

Execution node	-
Setting dialog	-

Equal Less

Input node names	InValue1 InValue2
Output node names	OutValue
Target node	-

Result	if (InValue1 <= InValue2)
	OutValue = “true”
	else
	OutValue = “false”

Execution node	-
Setting dialog	-

Not

Input node names	InValue
Output node names	OutValue
Target node	-
Result	OutValue = ! InValue
Execution node	-
Setting dialog	-

Object

SetLocation

Input node names	Location (x, y, z)
Output node names	-
Target node	Transform
Result	Transform : : location = location
Execution node	O
Setting dialog	O

SetRotation

Input node names	Rotation (x, y, z, angle)
Output node names	-
Target node	Transform
Result	Transform : : rotation = rotation
Execution node	O
Setting dialog	O

SetScale

Input node names	Scale (x, y, z)
Output node names	-
Target node	Transform
Result	Transform : : scale = Scale
Execution node	O
Setting dialog	O

SetCenter

Input node names	Center (x, y, z)
Output node names	-
Target node	Transform
Result	Transform : : center = center
Execution node	O
Setting dialog	O

GetLocation

Input node names	-
Output node names	Location (x, y, z)
Target node	Transform
Result	Location = Transform : : location
Execution node	-
Setting dialog	O

GetRotation

Input node names	-
Output node names	Rotation (x, y, z, angle)
Target node	Transform
Result	Rotation = Transform : : rotation
Execution node	-
Setting dialog	O

GetScale

Input node names	-
Output node names	Scale (x, y, z)
Target node	Transform
Result	Scale = Transform : : scale
Execution node	-
Setting dialog	O

GetCenter

Input node names	-
Output node names	Center (x, y, z)
Target node	Transform
Result	Center= Transform : : center
Execution node	-
Setting dialog	O

Material

SetAmbientIntensity

Input node names	AmbientIntensity
Output node names	-
Target node	Material
Result	Material : : ambientIntensity = AmbientIntensity
Execution node	O
Setting dialog	O

SetDiffuseColor

Input node names	DiffuseColor (r, g, b)
Output node names	-
Target node	Material
Result	Material : : diffuseColor = DiffuseColor
Execution node	O
Setting dialog	O

SetEmissiveColor

Input node names	EmissiveColor (r, g, b)
Output node names	-
Target node	Material
Result	Material : : emissiveColor = EmissiveColor
Execution node	O
Setting dialog	O

SetSpeculatColor

Input node names	SpecularColor (r, g, b)
Output node names	-
Target node	Material
Result	Material : : specularColor = SpecularColor
Execution node	O
Setting dialog	O

SetShininess

Input node names	Shininess
Output node names	-
Target node	Material
Result	Material : : shininess = Shininess
Execution node	O
Setting dialog	O

GetAmbientIntensity

Input node names	-
Output node names	AmbientIntensity
Target node	Material
Result	AmbientIntensity = Material : : ambientIntensity
Execution node	-
Setting dialog	O

GetDiffuseColor

Input node names	-
Output node names	DiffuseColor (r, g, b)
Target node	Material
Result	DiffuseColor = Material : : diffuseColor
Execution node	-
Setting dialog	O

GetEmissiveColor

Input node names	-
Output node names	EmissiveColor (r, g, b)
Target node	Material
Result	EmissiveColor = Material : : emissiveColor
Execution node	-
Setting dialog	O

GetSpecularColor

Input node names	-
Output node names	SpecularColor (r, g, b)
Target node	Material
Result	SpecularColor = Material : : specularColor
Execution node	-
Setting dialog	O

GetShininess

Input node names	-
Output node names	Shininess
Target node	Material
Result	Shininess= Material : : shininess
Execution node	-
Setting dialog	O

Light

SetOn

Input node names	On (“true” or “false”)
Output node names	-
Target node	DirectionalLight / PointLight / SpotLight
Result	Light : : on = On
Execution node	O
Setting dialog	O

SetColor

Input node names	Color (r, g, b)
Output node names	-
Target node	DirectionalLight / PointLight / SpotLight
Result	Light : : color = Color
Execution node	O
Setting dialog	O

SetIntensity

Input node names	Intensity
Output node names	-
Target node	DirectionalLight / PointLight / SpotLight
Result	Light : : intensity = Intensity
Execution node	O
Setting dialog	O

SetLocation

Input node names	Location (x, y, z)
Output node names	-
Target node	PointLight / SpotLight
Result	Light : : location= Location
Execution node	O
Setting dialog	O

SetDirection

Input node names	Direction (x, y, z)
Output node names	-
Target node	DirectionalLight / SpotLight
Result	Light : : intensity = Intensity
Execution node	O
Setting dialog	O

SetRadius

Input node names	Radius
Output node names	-
Target node	PointLight / SpotLight
Result	Light : : radius = Radius
Execution node	O
Setting dialog	O

GetOn

Input node names	-
Output node names	On (“true” or false)
Target node	DirectionalLight / PointLight / SpotLight
Result	On = Light : : on
Execution node	-
Setting dialog	O

GetColor

Input node names	-
Output node names	Color (r, g, b)
Target node	DirectionalLight / PointLight / SpotLight
Result	Color = Light : : color
Execution node	-
Setting dialog	O

GetIntensity

Input node names	-
Output node names	Intensity
Target node	DirectionalLight / PointLight / SpotLight
Result	Intensity = Light : : intensity
Execution node	-
Setting dialog	O

GetLocation

Input node names	-
Output node names	Location (x, y, z)
Target node	PointLight / SpotLight
Result	Location = Light : : location
Execution node	-
Setting dialog	O

GetDirection

Input node names	-
Output node names	Direction (x, y, z)
Target node	DirectionalLight / SpotLight
Result	Intensity = Light : : intensity
Execution node	-
Setting dialog	O

SetRadius

Input node names	-
Output node names	Radius
Target node	PointLight / SpotLight
Result	Radius = Light : : radius
Execution node	-
Setting dialog	O

Viewpoint

SetPosition

Input node names	Position (x, y, z)
Output node names	-
Target node	Viewpoint
Result	Viewpoint : : position = Position
Execution node	O
Setting dialog	O

SetOrientation

Input node names	Orientation (x, y, z, angle)
Output node names	-
Target node	Viewpoint
Result	Viewpoint : : orientaton = Orientation
Execution node	O
Setting dialog	O

SetFOV

Input node names	fov
Output node names	-
Target node	Viewpoint
Result	Viewpoint : : fieldOfView = fov
Execution node	O
Setting dialog	O

GetPosition

Input node names	-
Output node names	Position (x, y, z)
Target node	Viewpoint
Result	Position = Viewpoint : : position
Execution node	-
Setting dialog	O

GetOrientation

Input node names	-
Output node names	Orientation (x, y, z, angle)
Target node	Viewpoint
Result	Orientation = Viewpoint : : orientaton
Execution node	O
Setting dialog	O

GetFOV

Input node names	
Output node names	fov
Target node	Viewpoint
Result	fov = Viewpoint : : fieldOfView
Execution node	O
Setting dialog	O

Misc

GetTime

Input node names	-
Output node names	Hour Minute Second
Target node	-

Result	Hour = current system hour
	Minute = current system minute
	Second = current system second

Execution node	-
Setting dialog	-

Random

Input node names	-
Output node names	RandomValue
Target node	-
Result	RandomValue = 0.0 – 1.0
Execution node	-
Setting dialog	-

Beep

Input node names	-
Output node names	-
Target node	-
Result	Play a beep sound
Execution node	-
Setting dialog	-

JavaConsole

Input node names	String
Output node names	-
Target node	-
Result	Output the String into Java Console
Execution node	-
Setting dialog	-