Activity simulation engine

On this page

Cameo Simulation Toolkit provides an Activity simulation engine that allows you to run an Activity Simulation on Activity diagrams or Activity Elements. Cameo Simulation Toolkit also includes the implementation of OMG Semantics of a Foundational Subset for Executable UML Models (fUML), an executable subset of standard UML, that can be used to define the structural and Behavioral semantics of systems. fUML defines a basic virtual machine for the Unified Modeling Language and supports specific abstractions enabling compliant models to be transformed into various executable forms for verification, integration, and deployment.

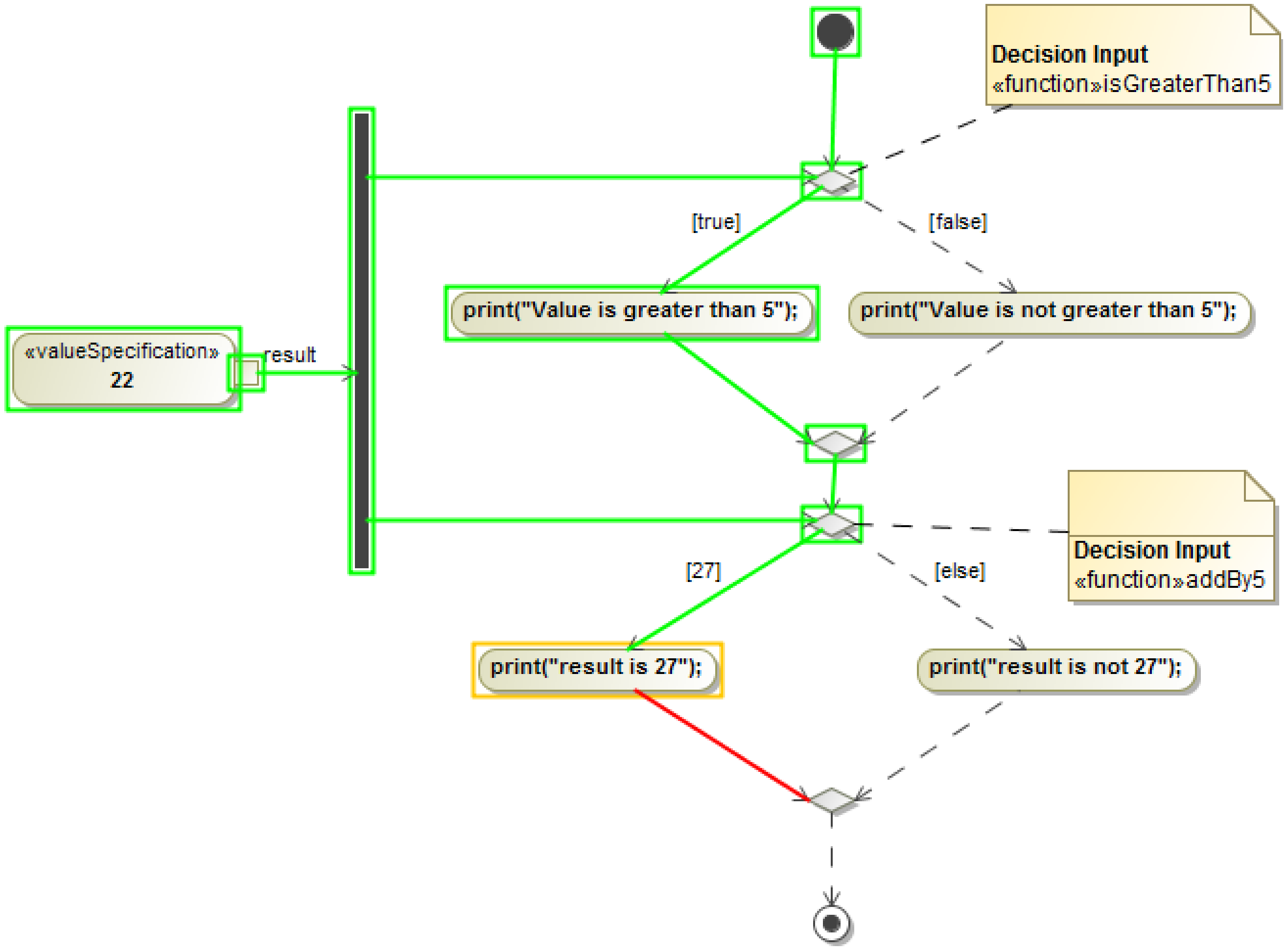
Various UML Activity diagram concepts are supported, including Object and Control Flows, Behavior and Operation Calls, sending Signals via Connectors with or without Ports in Internal Structure, accepting Signals and Time Events, Pins, Parameters, Decisions, Structured Activity Nodes, and many more.

The Activity simulation engine features include the following

- fUML 1.3 specification support.
- Any Action languages in opaqueBehaviors, opaqueExpressions, Decisions, Guards, and Constraints (see Integration with MATLAB® for more details).
- CallBehaviorAction with nested diagrams simulation and animation.
- SendSignalAction to send a Signal to a global Event queue to be used by the system level but not by the subsystem, e.g., in State Machine.

> ⚠ **Note**
>
> ○ SendSignalAction has the **Target** input Pin for specifying the signal receiver. If this Pin is omitted, the signal instance will be sent to the context itself.
>
> ○ The **On Port** property on SendSignalAction is used for sending signals via the specified port.

- CallOperationAction through a Port.
- Sending Signals through a Port.
- Support for Decision Nodes with probabilities over all outgoing edges.
- Support for Decision Nodes with a Decision input that provides input to Guard specifications on outgoing edges from each Decision Node.



The supported Decision input for Decision nodes.

Note

- You can simulate only Activities that are owned by a Package or a Class. As a workaround, the CallBehavior Actions, owned by the Call Behaviors in a Package, will be used for the entry/do/exit Behaviors in States.
- The Guards on an ObjectFlow are not Boolean expressions in fUML. They should contain a value that matches the runtime value that flows on the ObjectFlow during simulation.

To change to a regular UML (Boolean expression)

1. On the main menu, click **Options** > **Environment** and select **Simulation** on the left of the **Environment Options** dialog.
2. Select the **Use fUML DecisionSemantics value** check box so that the value becomes false. The value is false by default in the UML mode.

- A decision value can be passed as a parameter of Behavior of an opaque expression through a pin of Action. If a valid return value is from an assigned Behavior of a Guard, the body of the Guard will be ignored. However, if the Behavior is <unspecified>, the body of the Guard will be used.

Most of the elements on an Activity diagram are supported as follows

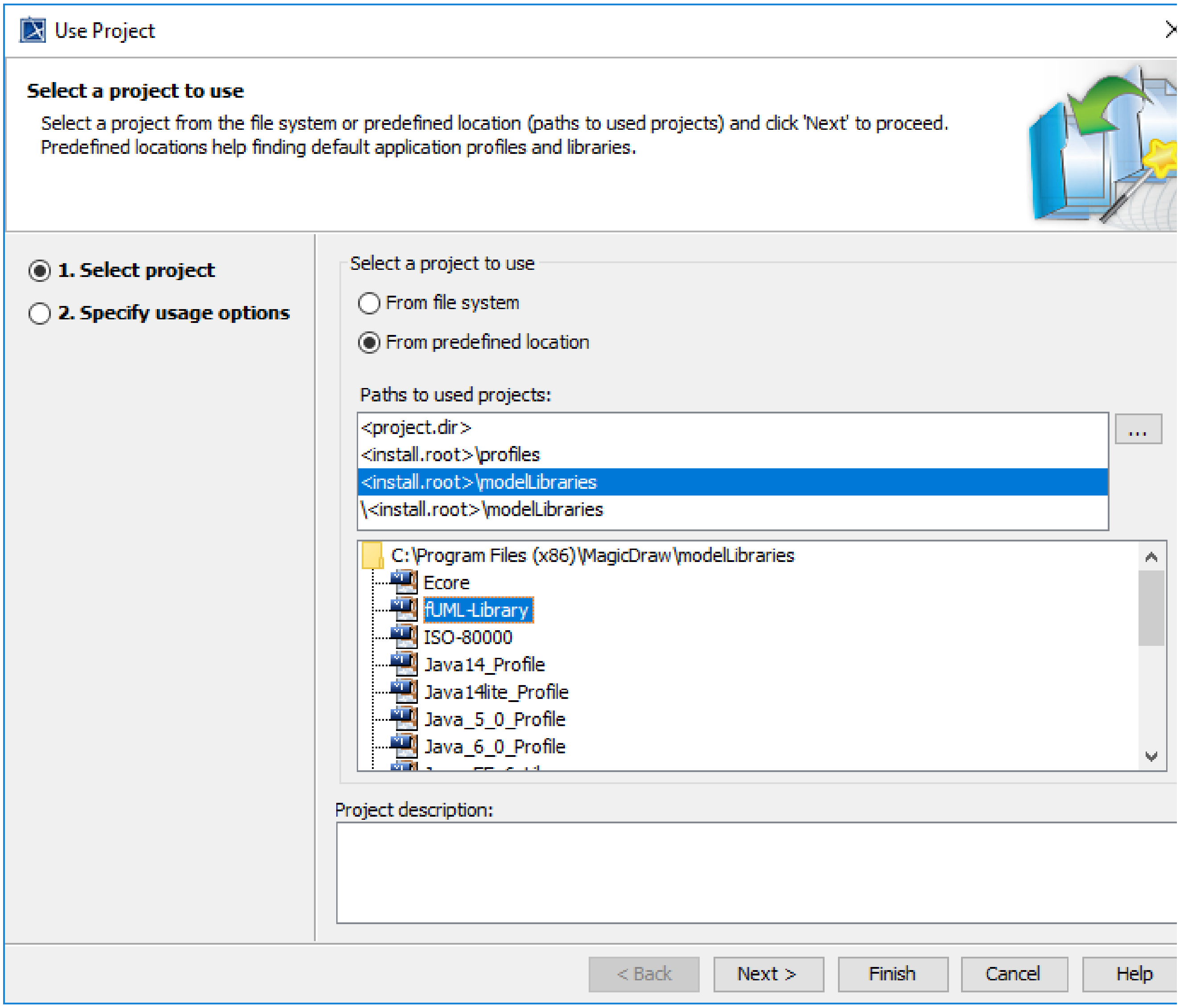
- Control Flow
- Object Flow
- Input Pin
- Output Pin
- Activity Final Node
- Flow Final Node
- Activity Parameter Node
- Decision Node
- Merge Node
- Join Node
- Fork Node
- Structured Activity Node
- Conditional Node
- Loop Node
- Expansion Region
- Expansion Node
- Object Node
  - Central Buffer Node
  - Data Store Node
- Actions
  - AcceptEventAction
  - AddStructuralFeatureValueAction
  - CallBehaviorAction
  - CallOperationAction
  - ClearAssociationAction
  - ClearStructuralFeatureAction
  - CreateLinkAction
  - CreateObjectAction
  - DestroyLinkAction
  - DestroyObjectAction
  - OpaqueAction
  - ReadExtentAction
  - ReadIsClassifiedObjectAction
  - ReadLinkAction
  - ReadSelfAction
  - ReadStructuralFeatureAction
  - ReclassifyObjectAction
  - ReduceAction
  - RemoveStructuralFeatureValueAction
  - SendSignalAction
  - StartClassifierBehaviorAction
  - StartObjectBehaviorAction
  - TestIdentityAction
  - ValueSpecificationAction

ReadLine support

ReadLine is a function that allows the user to enter value through the input line on the **Console** pane. A Call Behavior Action can be set Behavior as **Read Line [fUML_Library::BasicInputOutput]** using **fUML_Library.mdzip** from the **Use Project** dialog. Before using the ReadLine function, you need to include **fUML_Library.mdzip** in the project first.

To open the **Use Project** dialog and include **fUML_Library.mdzip**

1. Click **File** > **Use Project** > **Use Local Project** from the main menu to open the **Use Project** dialog.



Use Project

Select a project to use

Select a project from the file system or predefined location (paths to used projects) and click 'Next' to proceed.
Predefined locations help finding default application profiles and libraries.

● **1. Select project**

○ **2. Specify usage options**

Select a project to use

○ From file system

● From predefined location

Paths to used projects:

<project.dir>
<install.root>\profiles
<install.root>\modelLibraries
\<install.root>\modelLibraries

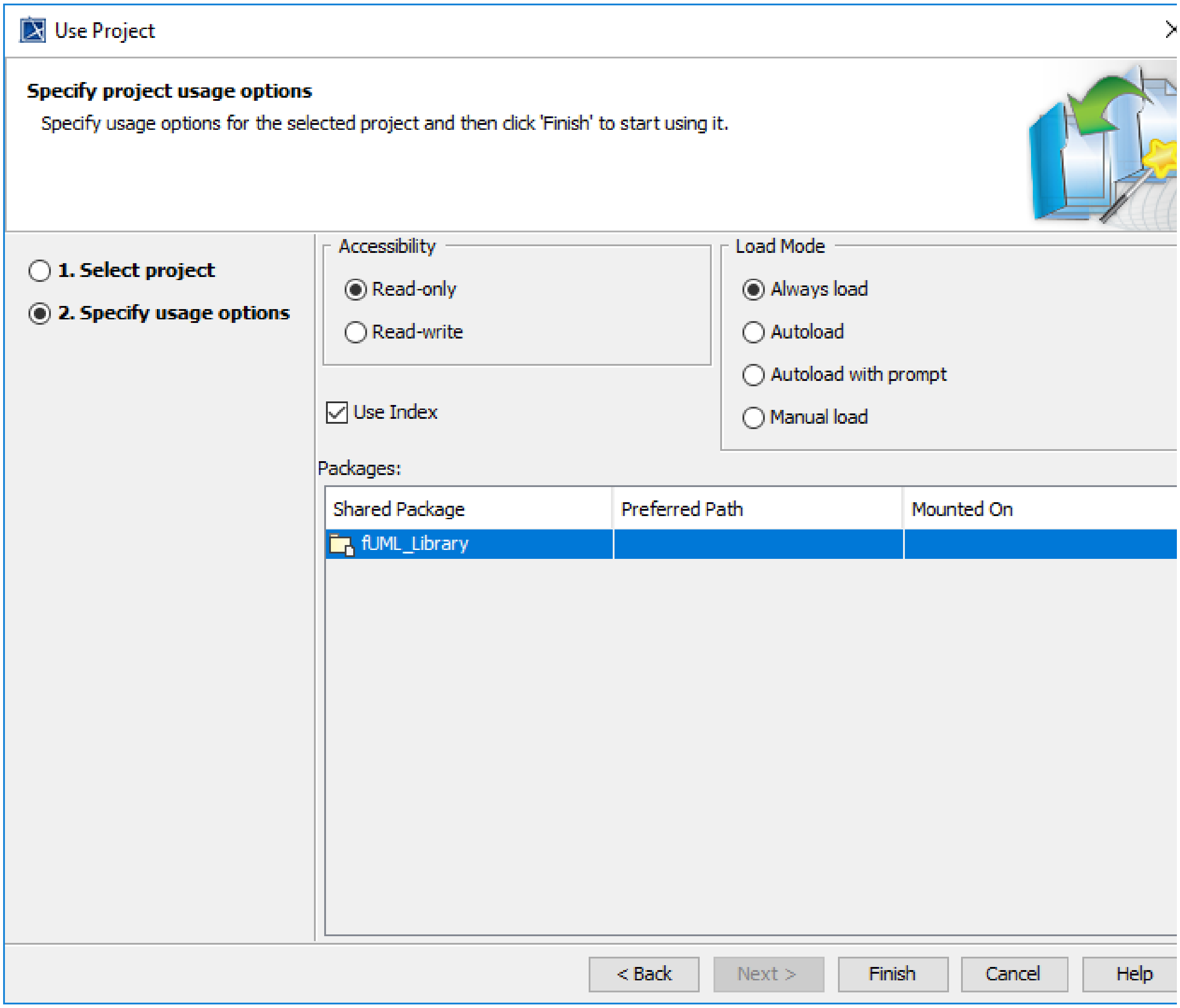
C:\Program Files (x86)\MagicDraw\modelLibraries
Ecore
fUML-Library
ISO-80000
Java14_Profile
Java14lite_Profile
Java_5_0_Profile
Java_6_0_Profile

Project description:

2. In the **Select a project to use** area, select the **From predefined location** option.

3. In the **Paths to used projects** list, select **<install.root>\modelLibraries**.

4. In the directory tree list, select **fUML-Library** and click **Next>** to proceed to the next step.



Use Project

Specify project usage options

Specify usage options for the selected project and then click 'Finish' to start using it.

○ 1. Select project

● 2. Specify usage options

Accessibility

● Read-only

○ Read-write

☑ Use Index

Load Mode

● Always load

○ Autoload

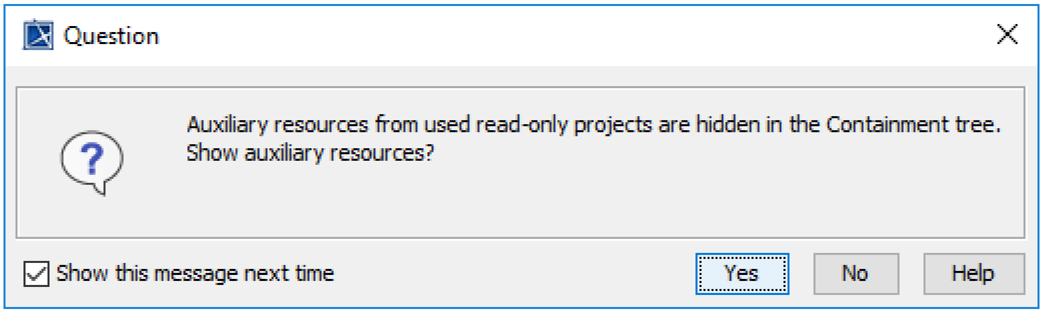
○ Autoload with prompt

○ Manual load

Packages:

Shared Package	Preferred Path	Mounted On
fUML_Library		

5. You will be at the **Specify usage options** step. Click **Finish**. The **Question** dialog opens to ask you about showing auxiliary resources in the Containment tree. Click **Yes**.

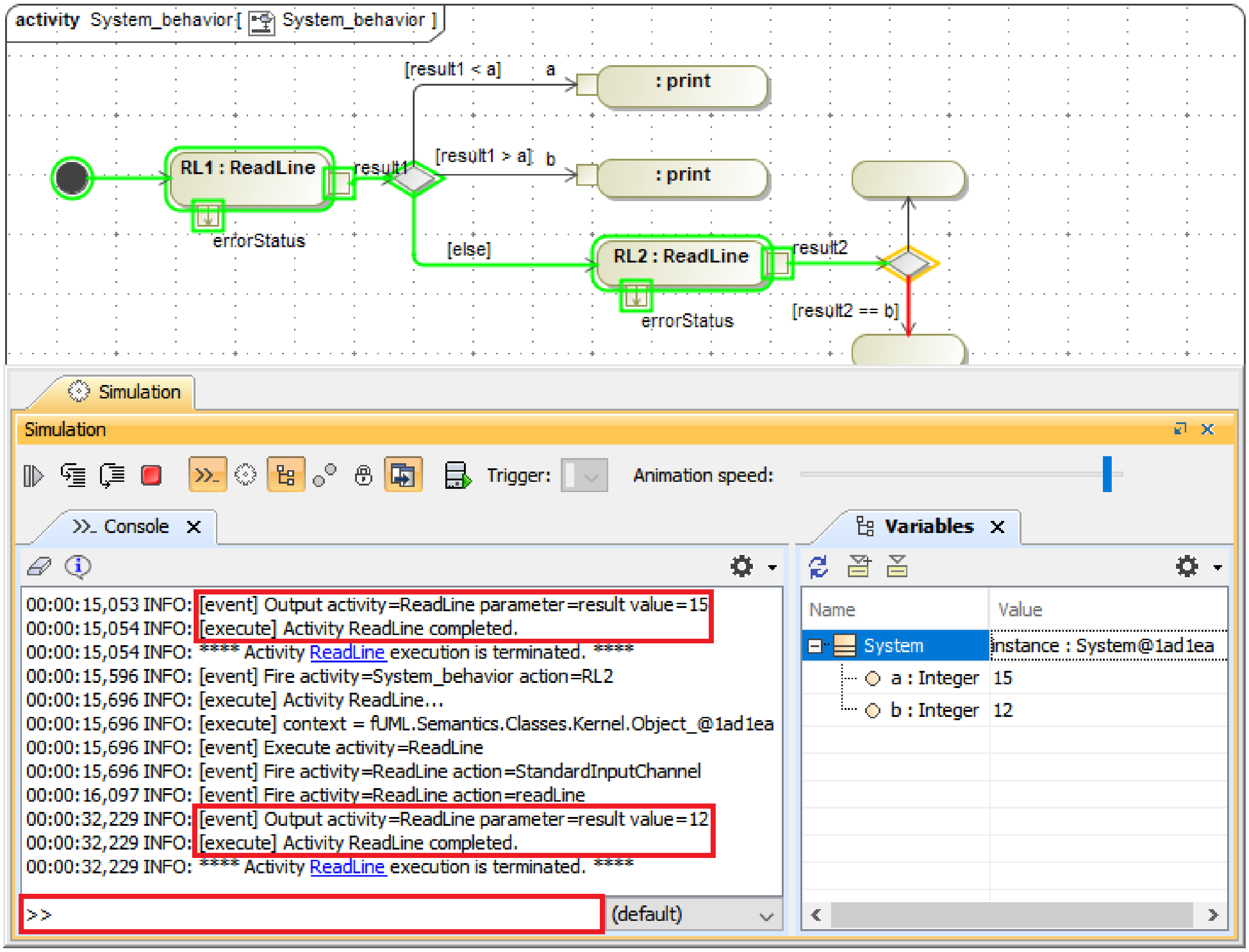


Question

Auxiliary resources from used read-only projects are hidden in the Containment tree. Show auxiliary resources?

☑ Show this message next time

Then a Call Behavior Action can now be set Behavior as a ReadLine Element. The ReadLine Element will be shown with two default Pins, i.e., result and errorStatus. During the simulation, the ReadLine Element is executed to allow entering value through the input line on the **Console** pane. The result of the ReadLine Element can be used by other Elements with any proper data types, e.g., Guard, as in the following figure



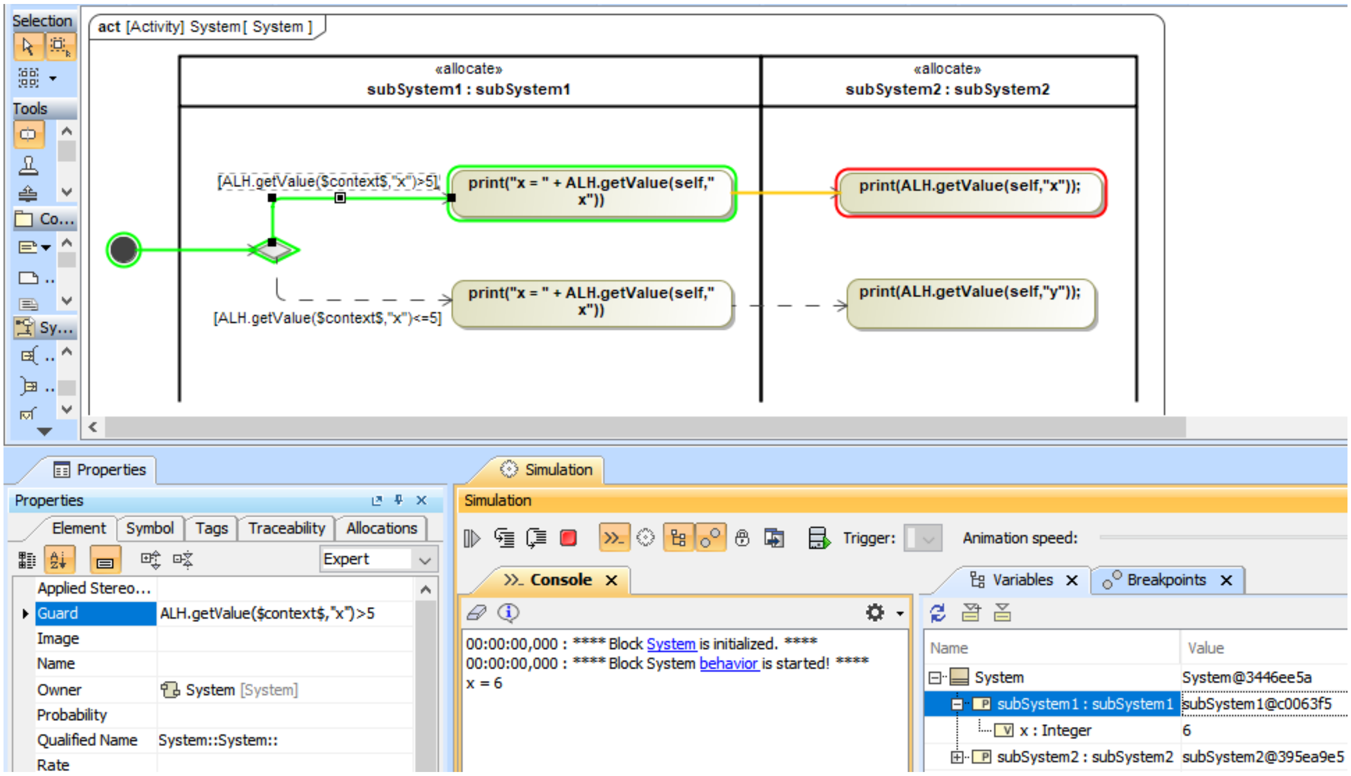
```
00:00:15,053 INFO: [event] Output activity=ReadLine parameter=result value=15
00:00:15,054 INFO: [execute] Activity ReadLine completed.
00:00:15,054 INFO: **** Activity ReadLine execution is terminated. ****
00:00:15,596 INFO: [event] Fire activity=System_behavior action=RL2
00:00:15,696 INFO: [execute] Activity ReadLine...
00:00:15,696 INFO: [execute] context = fUML.Semantics.Classes.Kernel.Object_@1ad1ea
00:00:15,696 INFO: [event] Execute activity=ReadLine
00:00:15,696 INFO: [event] Fire activity=ReadLine action=StandardInputChannel
00:00:16,097 INFO: [event] Fire activity=ReadLine action=readLine
00:00:32,229 INFO: [event] Output activity=ReadLine parameter=result value=12
00:00:32,229 INFO: [execute] Activity ReadLine completed.
00:00:32,229 INFO: **** Activity ReadLine execution is terminated. ****
>>
```

Name	Value
System	instance : System@1ad1ea
a : Integer	15
b : Integer	12

ReadLine support allows entering value through the input line on the Console pane.

Guards in Swimlanes

Simulation supports Guards in Swimlanes in the Activity diagram as shown in the figure below. See also Using Guards on Transitions and Swimlane.


Selection
Tools
Co...
Sy...
act [Activity] System[System]
«allocate»
subSystem1 : subSystem1
«allocate»
subSystem2 : subSystem2
[ALH.getValue($context$,"x")>5]
print("x = " + ALH.getValue(self," x"))
print(ALH.getValue(self,"x"));
[ALH.getValue($context$,"x")<=5]
print("x = " + ALH.getValue(self," x"))
print(ALH.getValue(self,"y"));
Properties
Element
Symbol
Tags
Traceability
Allocations
Expert
Simulation
Trigger:
Animation speed:
Console
Variables
Breakpoints

Applied Stereo...	
Guard	ALH.getValue($context$,"x")>5
Image	
Name	
Owner	System [System]
Probability	
Qualified Name	System::System::
Rate	

```
00:00:00,000 : **** Block System is initialized. ****
00:00:00,000 : **** Block System behavior is started! ****
x = 6
```

Name	Value
System	System@3446ee5a
subSystem1 : subSystem1	subSystem1@c0063f5
x : Integer	6
subSystem2 : subSystem2	subSystem2@395ea9e5

Guards in Swimlanes are supported in the Activity diagram.

Selecting context objects

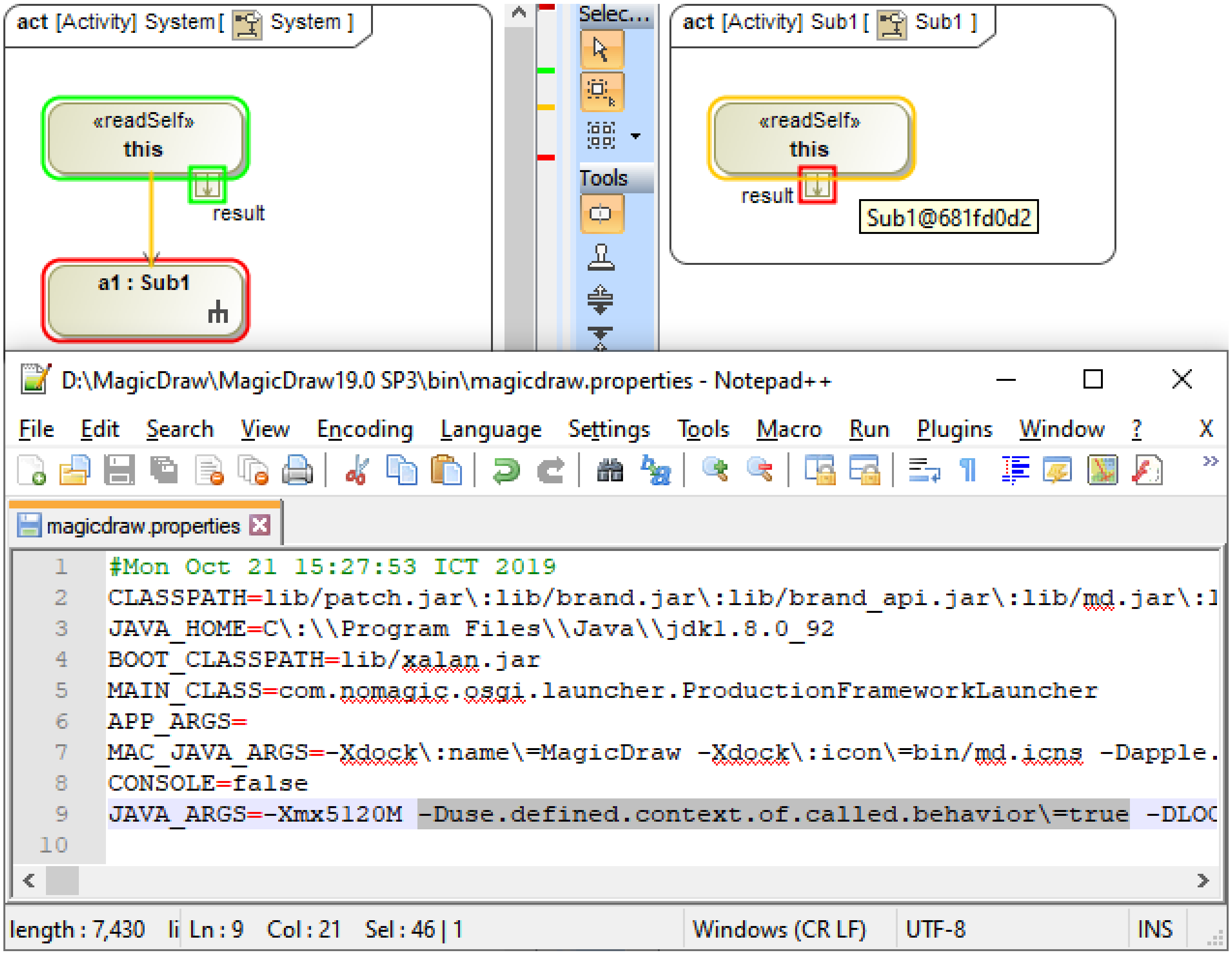
You can select context objects by adding the following Java property to the property file of the modeling tool, e.g., **magicdraw.properties** or **csm.properties**, on the **JAVA_ARGS** line.

```
-Duse.defined.context.of.called.behavior\=true
```

Note

- The **-Duse.defined.context.of.called.behavior\=true** Java property is not included in the product by default.
- To turn off the Java property for selecting context objects, set the property **=false** or remove the property.

From the figure below, the *readSelf* Action places the context object of the Action execution on its result at the Output Pin using the option for selecting the context object. The result of the Output Pin is shown with the *Sub1* object. If the option for the context object is not used, the result of the Output Pin will be shown with the *System* object instead.


act [Activity] System [System]
«readSelf»
this
result
a1 : Sub1
Selec...
Tools
act [Activity] Sub1 [Sub1]
«readSelf»
this
result
Sub1@681fd0d2
D:\MagicDraw\MagicDraw19.0 SP3\bin\magicdraw.properties - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
magicdraw.properties
length : 7,430 li Ln : 9 Col : 21 Sel : 46 | 1 Windows (CR LF) UTF-8 INS

```
#Mon Oct 21 15:27:53 ICT 2019
CLASSPATH=lib/patch.jar\:lib/brand.jar\:lib/brand_api.jar\:lib/md.jar\:l
JAVA_HOME=C\:\\Program Files\\Java\\jdk1.8.0_92
BOOT_CLASSPATH=lib/xalan.jar
MAIN_CLASS=com.nomagic.osgi.launcher.ProductionFrameworkLauncher
APP_ARGS=
MAC_JAVA_ARGS=-Xdock\:name\=MagicDraw -Xdock\:icon\=bin/md.icns -Dapple.
CONSOLE=false
JAVA_ARGS=-Xmx5120M -Duse.defined.context.of.called.behavior\=true -DLOC

```

The result of the Output Pin shown with the context object (Sub1) according to the Java property setting.

Related pages

- Decision and Merge
- Behavior
- Action