



User Manual

Software Version: 2017-08
July 20, 2017

© 2017 JaamSim Software Inc.

Table of Contents

| | | |
|-----------|-----------------------------------------------------|-----------|
| 1 | Introduction | 1 |
| 2 | Installing and Running JaamSim | 2 |
| 2.1 | System Requirements | 2 |
| 2.2 | Installing Java..... | 2 |
| 2.3 | Installing JaamSim | 2 |
| 2.4 | Running JaamSim from the GUI..... | 3 |
| 2.5 | Running JaamSim from the Command Line..... | 3 |
| 3 | JaamSim Basic Example | 5 |
| 3.1 | Step 1: Creating Model Objects..... | 6 |
| 3.2 | Step 2: Putting the Objects Together | 8 |
| 3.3 | Step 3: Adding a Probability Distribution | 10 |
| 3.4 | Step 4: Changing Model Graphics..... | 12 |
| 4 | Graphical User Interface..... | 13 |
| 4.1 | Control Panel..... | 13 |
| 4.2 | View Windows | 17 |
| 4.3 | Model Builder | 21 |
| 4.4 | Object Selector..... | 22 |
| 4.5 | Input Editor | 23 |
| 4.6 | Output Viewer..... | 25 |
| 5 | Units | 26 |
| 5.1 | Unit Types | 26 |
| 5.2 | Defining a New Unit..... | 27 |
| 6 | Expressions and User-Defined Variables | 28 |
| 6.1 | Expressions..... | 28 |
| 6.2 | User-Defined Variables | 36 |
| 7 | Simulation Runs and Experiments | 38 |
| 7.1 | Simulation Object | 38 |
| 7.2 | Performing Multiple Simulation Runs | 40 |
| 7.3 | Customized Output Report..... | 41 |
| 8 | Configuration File | 42 |
| 8.1 | Basic Structure | 42 |
| 8.2 | Object Definitions | 42 |
| 8.3 | Object Inputs | 42 |
| 8.4 | Include Statements..... | 43 |
| 8.5 | Groups | 43 |
| 8.6 | RecordEdits Statement | 44 |
| 8.7 | Example Configuration File | 44 |
| 9 | Maintenance, Breakdowns, and Thresholds..... | 47 |
| 9.1 | Thresholds..... | 47 |
| 9.2 | Maintenance and Breakdowns | 48 |
| 9.3 | States | 49 |
| 10 | Graphics | 51 |
| 10.1 | DisplayEntity..... | 51 |
| 10.2 | DisplayModel..... | 52 |
| 10.3 | Importing a 3D Object or Image | 56 |
| 10.4 | View Window..... | 57 |
| 11 | Graphics Objects Palette..... | 58 |
| 11.1 | Region..... | 59 |
| 11.2 | DisplayEntity..... | 60 |

| | | |
|-----------|----------------------------------------------------------------|------------|
| 11.3 | Text | 61 |
| 11.4 | EntityLabel..... | 63 |
| 11.5 | InputBox | 64 |
| 11.6 | BarGauge..... | 65 |
| 11.7 | Overlay Objects (OverlayImage, OverlayText, OverlayClock)..... | 66 |
| 11.8 | BillboardText | 66 |
| 11.9 | Arrow | 67 |
| 11.10 | Graph | 68 |
| 11.11 | MimicEntity | 71 |
| 11.12 | VideoRecorder | 72 |
| 12 | Probability Distributions Palette..... | 74 |
| 12.1 | Changing the Random Seed | 75 |
| 12.2 | UniformDistribution..... | 76 |
| 12.3 | TriangularDistribution | 77 |
| 12.4 | NormalDistribution | 78 |
| 12.5 | ExponentialDistribution..... | 79 |
| 12.6 | NonStatExponentialDist | 80 |
| 12.7 | ErlangDistribution | 81 |
| 12.8 | GammaDistribution..... | 82 |
| 12.9 | BetaDistribution | 83 |
| 12.10 | WeibullDistribution..... | 84 |
| 12.11 | LogNormalDistribution | 85 |
| 12.12 | LogLogisticDistribution | 86 |
| 12.13 | DiscreteDistribution | 87 |
| 12.14 | ContinuousDistribution | 88 |
| 12.15 | BooleanSelector | 89 |
| 13 | Basic Objects Palette..... | 90 |
| 13.1 | InputValue | 91 |
| 13.2 | TimeSeries | 92 |
| 13.3 | TimeSeriesThreshold | 94 |
| 13.4 | ExpressionThreshold..... | 96 |
| 13.5 | BooleanIndicator | 98 |
| 13.6 | ExpressionLogger | 99 |
| 13.7 | EntitlementSelector | 101 |
| 13.8 | ExpressionEntity..... | 102 |
| 13.9 | DowntimeEntity | 103 |
| 13.10 | ValueSequence | 105 |
| 13.11 | EventSchedule | 106 |
| 13.12 | FileToVector and FileToMatrix | 107 |
| 13.13 | ScriptEntity | 110 |
| 14 | Process Flow Palette | 111 |
| 14.1 | SimEntity | 113 |
| 14.2 | EntityLauncher | 114 |
| 14.3 | EntityGenerator | 115 |
| 14.4 | EntitySink | 117 |
| 14.5 | Server..... | 118 |
| 14.6 | Queue | 120 |
| 14.7 | EntityConveyor | 122 |
| 14.8 | EntityDelay | 124 |
| 14.9 | Resource..... | 125 |
| 14.10 | Seize | 127 |
| 14.11 | Release | 129 |
| 14.12 | Assign | 130 |
| 14.13 | Branch..... | 132 |
| 14.14 | Duplicate | 133 |
| 14.15 | Combine..... | 134 |

| | | |
|-----------|------------------------------------------|------------|
| 14.16 | SetGraphics..... | 136 |
| 14.17 | EntityGate..... | 137 |
| 14.18 | EntitySignal | 139 |
| 14.19 | SignalThreshold | 140 |
| 14.20 | Assemble | 141 |
| 14.21 | EntityContainer..... | 143 |
| 14.22 | Pack | 144 |
| 14.23 | Unpack | 146 |
| 14.24 | AddTo..... | 148 |
| 14.25 | RemoveFrom..... | 150 |
| 14.26 | EntityLogger | 152 |
| 14.27 | Statistics..... | 153 |
| 15 | Calculation Objects Palette | 155 |
| 15.1 | Controller..... | 156 |
| 15.2 | WeightedSum..... | 157 |
| 15.3 | Polynomial..... | 158 |
| 15.4 | Integrator..... | 159 |
| 15.5 | Differentiator..... | 160 |
| 15.6 | PIDController..... | 161 |
| 15.7 | Lag | 163 |
| 15.8 | MovingAverage | 164 |
| 15.9 | SineWave..... | 165 |
| 15.10 | SquareWave..... | 166 |
| 15.11 | UnitDelay..... | 167 |
| 16 | Fluid Objects Palette..... | 168 |
| 16.1 | Fluid | 169 |
| 16.2 | FluidFlow..... | 170 |
| 16.3 | FluidFixedFlow | 171 |
| 16.4 | FluidTank | 172 |
| 16.5 | FluidPipe | 173 |
| 16.6 | FluidCentrifugalPump..... | 174 |

Appendices

A Named Colours

1 Introduction

JaamSim (Java Animation Modelling and Simulation) is a discrete-event simulation software package first developed in 2002 as the foundation for simulation applications. JaamSim includes a drag-and-drop graphical user interface, 3D animation, and a full set of built-in objects for model building. It is object oriented, extremely fast, and scalable to the largest of applications. Windows, Linux, and OSX are all supported.

JaamSim is free open-source software, licensed under Apache 2.0. The latest version of the software and manuals can be downloaded from the JaamSim website: www.jaamsim.com. The source code is published on GitHub: www.github.com/jaamsim/jaamsim. Presentations and tutorials for JaamSim can be found by following the Videos link on the JaamSim website.

JaamSim provides all the key functions needed for any simulation model:

- Controls for launching and manipulating simulation runs;
- Drag-and-drop user interface;
- Interactive 3D graphics;
- Input and output processing; and
- Model development tools and editors.

JaamSim also provides a full suite of built-in objects for model building, including:

- Objects for process flow type models (servers, queues, etc.);
- Objects for modelling continuous processes (integrator, PID controller, etc.);
- Text objects for labelling and documentation;
- Graphs for visualizing simulation outputs;
- Probability distributions for random sampling; and
- Graphical objects for background maps and logos.

Advanced users can create additional palettes of application-specific objects. A separate Programming Manual can be downloaded from the JaamSim website.

2 Installing and Running JaamSim

2.1 System Requirements

JaamSim runs under Windows, Linux, and OSX on most modern computers. Any computer with an Intel Core i3, i5, and i7 series processor is sufficient for JaamSim (second generation "Sandy Bridge" processor and later).

For models with complex 3D graphics, a NVIDIA GeForce graphics card is recommended. Workstation-type graphics cards such as NVIDIA Quadro series also work well, but do not provide improved performance over less expensive GeForce cards. Although it is possible to use a computer with an AMD graphics card, NVIDIA cards generally provide better support for OpenGL graphics and are preferred for JaamSim.

2.2 Installing Java

JaamSim requires a recent (Version 7 or later) installation of the Java Runtime Environment (JRE), available for download free-of-charge from www.java.com.

The default JRE for Windows computers is the 32-bit version, in which case the 32-bit version of JaamSim must be used.

For computers running 64-bit Windows, the 64-bit version of JaamSim offers improved performance. To obtain the 64-bit version of the JRE, follow the link "See all Java downloads" and select "Windows Offline (64-bit)".

It is acceptable to install both the 32-bit and 64-bit JREs on the same computer.

2.3 Installing JaamSim

JaamSim consists of a single executable that can be copied directly to the user's computer. No special installation program is required. Copy the JaamSim executable file to a working directory, such as the directory that will contain the model input files.

Three versions of the JaamSim executable are available for each release:

- JaamSimYYYY-NN.exe (the 64-bit executable for Windows)
- JaamSimYYYY-NN_x86.exe (the 32-bit executable for Windows)
- JaamSimYYYY-NN.jar (the executable jar file for Windows, Mac OSX, and Linux)

JaamSim releases are denoted by the year of release (YYYY) and by release number (NN) within the year.

2.3.1 Installation Problems

If JaamSim does not run correctly on your computer, the most likely cause is an older graphics driver. Updating the driver to the latest version will solve the problem.

2.3.2 Mac Computers with the Retina Monitor

Mac computers with the Retina monitor require the use of the "Display Menu" app that can be downloaded from the Apple App Store. This app compensates for the non-standard way in which Apple uses the higher resolution of the Retina monitor. If JaamSim is launched without this app, it will not be possible to select objects using the mouse.

2.3.3 Ubuntu Linux

The Intel graphics driver for Ubuntu has a long-standing bug that affects JaamSim. It can be avoided by entering the following command before launching JaamSim:

```
export LIBGL_ALWAYS_SOFTWARE=1
```

This command instructs Ubuntu to use software rendering, which, although slower, is sufficient for most models.

2.4 Running JaamSim from the GUI

JaamSim can be launched by double-clicking on the executable file.

2.5 Running JaamSim from the Command Line

JaamSim can also be launched, configured and started automatically from the command line or a batch file using the command:

```
JaamSim.exe config1.cfg -tags
```

or, when using the .jar file:

```
java -jar JaamSim.jar config1.cfg -tags
```

Here, config1.cfg is the name of the input file to be loaded and -tags are the optional tags for the run. Multiple tags must be separated by a space. The following tags are supported:

Table 2-1 Batch Mode Run Tags

| Tag | Description |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -b or -batch | Starts the simulation immediately after the input file has been read, and exits when the run has completed. This tag is useful for batch file execution. |
| -m or -minimize | Minimizes the graphical user interface, allowing the simulation to run slightly faster when visualizations are not required (for instance, in overnight simulation runs). |
| -s or -script | Directs JaamSim to accept configuration file inputs piped to JaamSim through standard-in and to direct its outputs specified by the RunOutputList keyword to standard-out. The .jar file (jaamsim.jar) must be used with this feature, not the executable (jaamsim.exe). |
| -h or -headless | Runs JaamSim without the graphical user interface so that it can be executed on a server that has no graphics capability. Batch mode (-b) is set automatically with this option. |

It is also possible to load two or more configuration files into a single model using the following command:

```
JaamSim.exe config1.cfg config2.cfg -tags
```

or,

```
java -jar JaamSim.jar config1.cfg config2.cfg -tags
```

Multiple simulation runs can be executed one after the other by using a batch file that contains a series of these commands. For example, a batch file containing the following two lines would execute two runs: run1.cfg and run2.cfg:

```
JaamSim.exe run1.cfg -b  
JaamSim.exe run2.cfg -b
```

Note that the batch file and input configuration files must be in the same directory for this example to work.

JaamSim can be interfaced with other software packages using the -s (script) tag. For example, the following command instructs JaamSim to load the configuration file config.cfg and then accept additional configuration file inputs from program1. The outputs specified by the RunOutputList keyword for Simulation are then directed as inputs to program2.

```
program1.exe | java -jar JaamSim.jar config.cfg -s -b | program2.exe
```

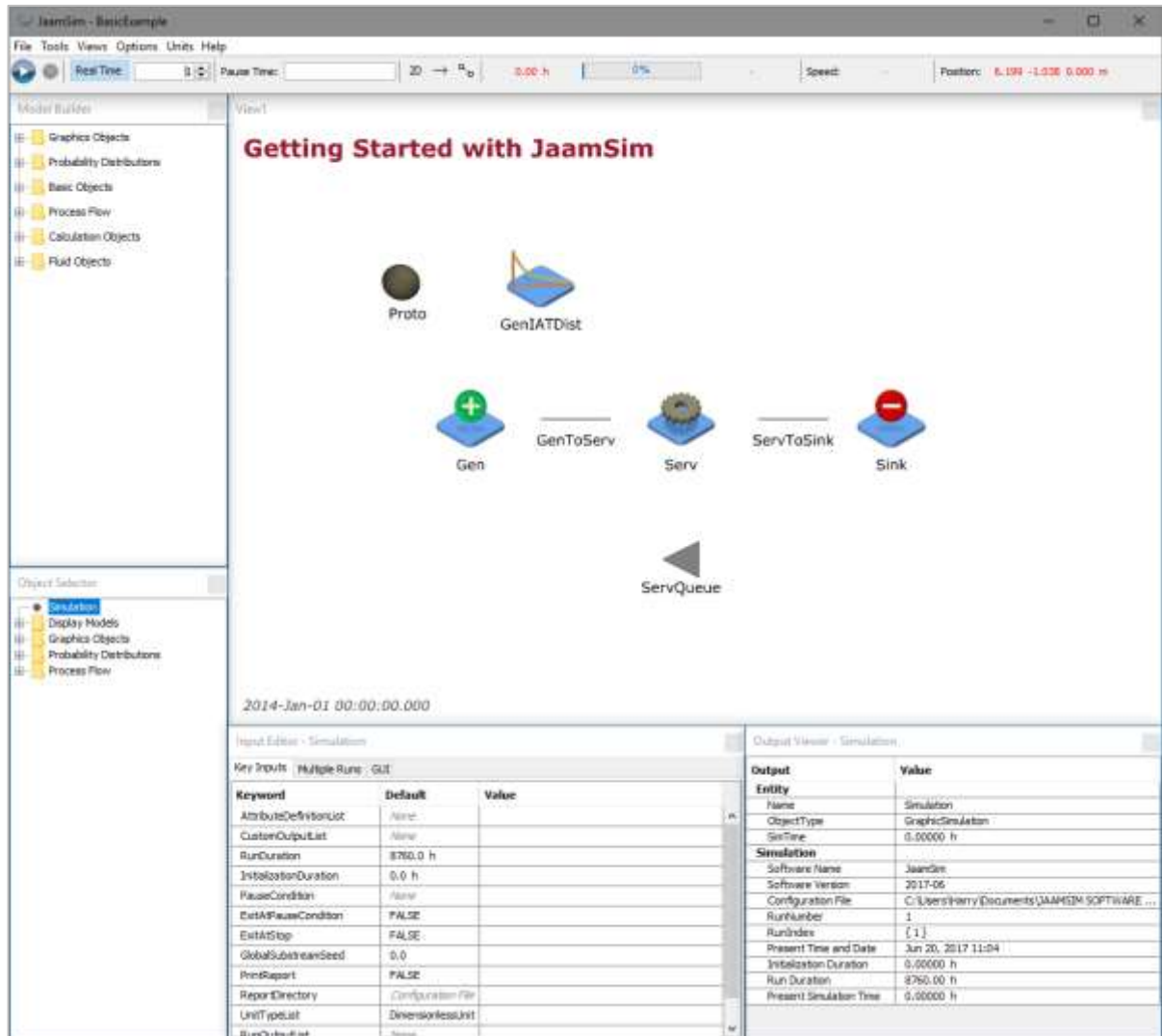
At present, the script tag is supported only for the .jar file version of JaamSim - it does not work with the .exe file version.

Note that "java -jar" must be used in this command for standard-in and standard-out to be connected correctly to JaamSim. Without the "java -jar" portion of the command, the java virtual machine sets both standard-in and standard-out to null.

3 JaamSim Basic Example

In this example, you will be guided through building a basic model using the graphical user interface (GUI). The example model simulates a typical single-server queuing system, with entities being generated, processed, and consumed. The finished model is shown below.

Figure 3-1 Screenshot of the JaamSim Basic Example



This example is divided into four steps:

- Step 1: Creating Model Objects
- Step 2: Putting the Object Together
- Step 3: Changing Model Graphics
- Step 4: Adding a Probability Distribution

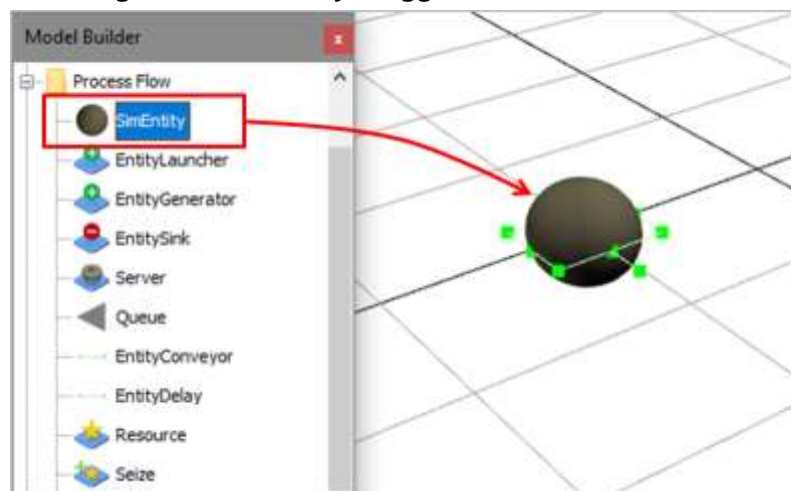
3.1 Step 1: Creating Model Objects

After launching JaamSim, the following windows will appear:

- Control Panel – provides a number of run control features;
- View Window – displays a graphical representation of the model;
- Model Builder – offers a selection of objects that can be added to the model;
- Object Selector – lists the objects present in the model;
- Input Editor – allows for editing of keywords for a selected object; and
- Output Viewer – displays outputs for a selected object.

In the Model Builder, expand the Process Flow palette and then drag-and-drop a SimEntity into the View Window (View1). This creates a SimEntity object with a default name (SimEntity1) and shape (Sphere) and automatically selects it, denoted by green highlighting as below.

Figure 3-2 SimEntity Dragged to the View Window



This object will serve as the prototype for entities that will be processed in the model. In the Object Selector, select SimEntity1 and press F2 to rename the object as 'Proto'.

Table 3-1 SimEntity Object to Create

| Model Builder Palette | Object Type | Name |
|-----------------------|-------------|-------|
| Process Flow | SimEntity | Proto |

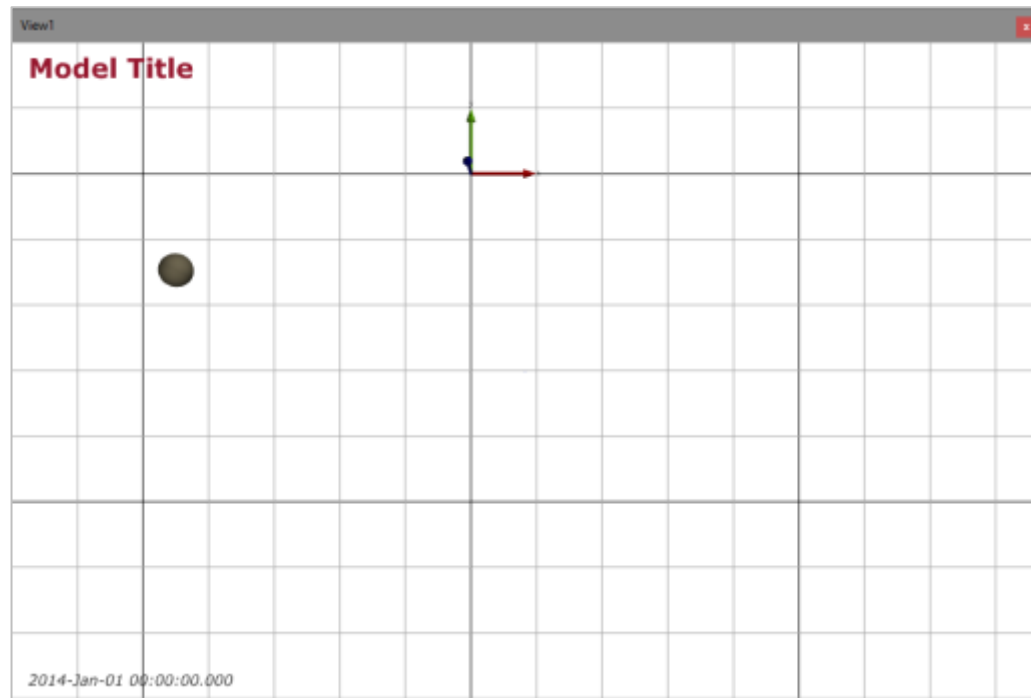
Since the model graphics will be 2D, aside from the Proto entity, click the '2D' button on the Control Panel, so that the view becomes bird's eye. Adjust the viewing position using the actions listed below.

Table 3-2 View Adjustments

| Action | Description |
|-------------------|---------------------|
| Left Click + Drag | Pan in the XY-plane |
| Scroll Wheel | Zoom in or out |

After adjusting the view position and zoom, the View window should appear similar the following figure.

Figure 3-3 Configured View Window



Now create and rename the objects listed below:

Table 3-3 Additional Objects to Create

| Model Builder Palette | Object Type | Name |
|-----------------------|-----------------|------------|
| Process Flow | EntityGenerator | Gen |
| Process Flow | EntityConveyor | GenToServ |
| Process Flow | Server | Serv |
| Process Flow | EntityConveyor | ServToSink |
| Process Flow | EntitySink | Sink |
| Process Flow | Queue | ServQueue |

It is often easier to rename an object by turning on its label, which can then be edited. Start by right-click on the object and selecting 'Show Label'. Then double-click on the label and enter the new name. Changing an object's label causes it to be renamed accordingly. Renaming is completed by pressing the Return button or by clicking elsewhere in the View window. The label can be left in place or it can be turned off by right-clicking on the object and de-selecting 'Show Label'.

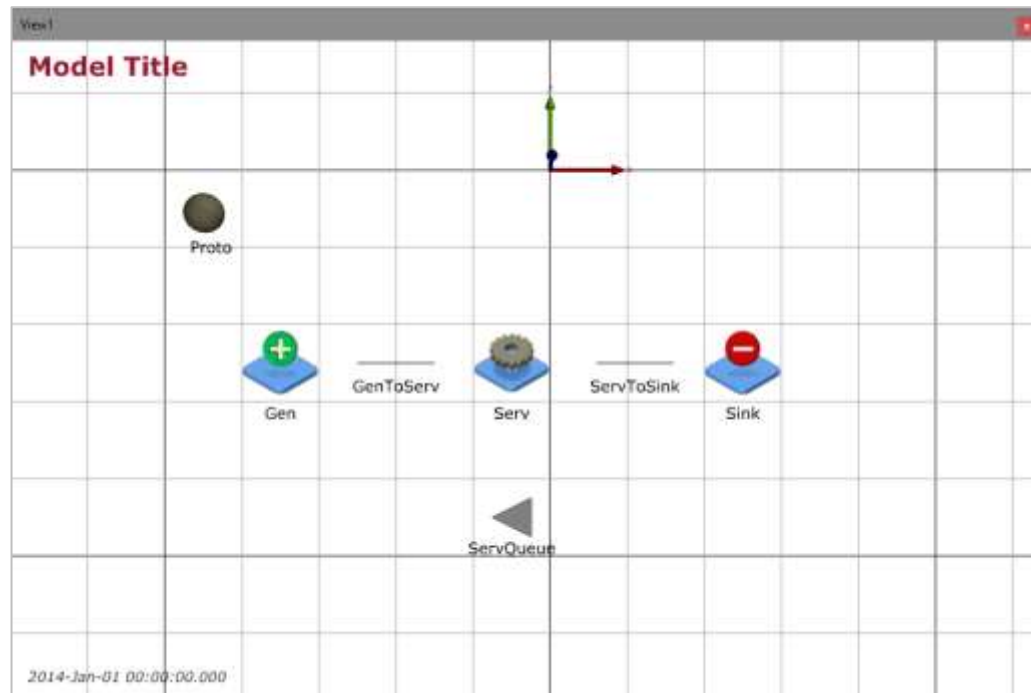
An object can be moved by selecting it and using CTRL + Left Click + Drag. The position of a label relative to its object can be changed using the same method. Position the first five objects from left to right in the following order:

Gen - GenToServ - Serv - ServToSink - Sink

Place the final object, ServQueue, below the Serv object.

After positioning the objects, the model should look similar to the following figure.

Figure 3-4 Screenshot of Step 1



3.2 Step 2: Putting the Objects Together

In this step, the objects placed in Step 1 must be set to interact with one another. Connect the objects together by setting the keyword values listed below. Connect the objects together by setting the keyword values listed below.

Table 3-4 Object Connections

| Object | Keyword | Value |
|------------|-----------------|------------|
| Gen | PrototypeEntity | Proto |
| Gen | NextComponent | GenToServ |
| GenToServ | NextComponent | ServQueue |
| Serv | WaitQueue | ServQueue |
| Serv | NextComponent | ServToSink |
| ServToSink | NextComponent | Sink |

The PrototypeEntity input tells the EntityGenerator to make copies of the SimEntity named Proto. The NextComponent input tells each object where to send the generated SimEntities after it has finished its processing. The WaitQueue input tells the Server to store its waiting SimEntities in the Queue named ServQueue.

Model inputs are normally set using the Input Editor. However, for the setting inputs that specify the flow of entities through a model, it is often more convenient to use the 'Create Entity Links' and 'Show Entity Flow' buttons on the Control Panel. Depressing the 'Show Entity Flow' button will display an

arrow between objects as the flow connections are made. Depressing the 'Create Entity Links' button allows the user to make the above connections by clicking on each object in the order in which the SimEntities flow through the model. In this case, click on the objects in the following sequence:

Proto – Gen – GenToServ – ServQueue – Serv – ServToSink - Sink

If you make a mistake, you can interrupt the connection process by clicking on the background, and start again with the next object to be connected. When finished, click on the 'Create Entity Links' button again to de-activate it. You can confirm the connections by checking the relevant inputs using Input Editor.

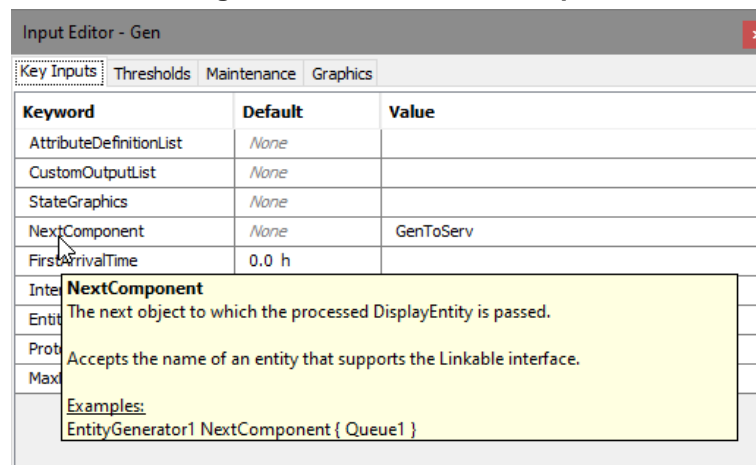
Now, use the Input Editor to set remaining inputs, i.e. the time at which Proto entities will be added to the model by Gen, and the time that entities will spent at each stage of the model.

Table 3-5 Transit and Handling Durations

| Object | Keyword | Value |
|------------|------------------|-------|
| Gen | InterArrivalTime | 2 s |
| GenToServ | TravelTime | 1 s |
| Serv | ServiceTime | 1 s |
| ServToSink | TravelTime | 1.5 s |

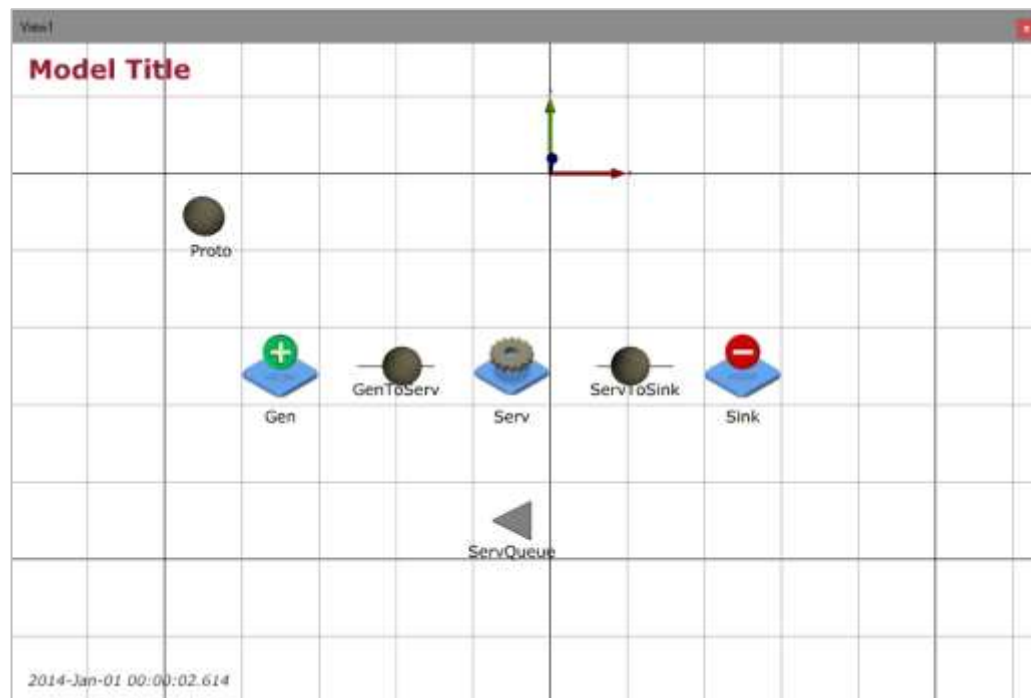
Hovering the cursor over keywords in the Input Editor will display a brief description of the keyword. For example, the mouse-over for the NextComponent keyword is shown in the following figure.

Figure 3-5 Mouse-Over Tooltips



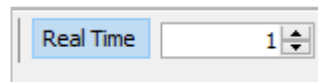
Save the model by selecting 'Save As...' from the File menu, and press the Play button to run the simulation. The model should appear similar to the following screenshot.

Figure 3-6 Screenshot of Step 2



Note that the Real Time button in the Control Panel is depressed, as shown below.

Figure 3-7 Real Time Controls



This indicates that the model speed is restricted to the Real Time speed multiplier shown in the text box to the right of the Real Time button. The Real Time speed multiplier controls how fast the simulated time elapses in the model. The default Real Time speed multiplier is 1, meaning that each real (wall-clock) second corresponds to one simulated second. It can be changed by entering the desired Real Time speed multiplier into the text box, or by pressing the up or down arrow buttons (which double or halve the Real Time speed multiplier factor respectively).

To continue, Pause or Reset the model by selecting the corresponding buttons on the Control Panel. Pausing the model allows the simulation to resume later starting from the paused time, while resetting the model forces the model to start from time zero.

3.3 Step 3: Adding a Probability Distribution

In this step, dynamic variability is added to the model by including a probability distribution to control the inter-arrival time of Proto entities.

Create an exponential distribution object as listed below.

Table 3-6 Object to Create for Step 3

| Model Builder Palette | Object Type | Name |
|---------------------------|-------------------------|------------|
| Probability Distributions | ExponentialDistribution | GenIATDist |

Set the keyword values for GenIATDist as shown below.

Table 3-7 GenIATDist Inputs

| Tab | Keyword | Value |
|------------|----------|----------|
| Key Inputs | UnitType | TimeUnit |
| Key Inputs | MinValue | 0 s |
| Key Inputs | MaxValue | 10 s |
| Key Inputs | Mean | 2 s |

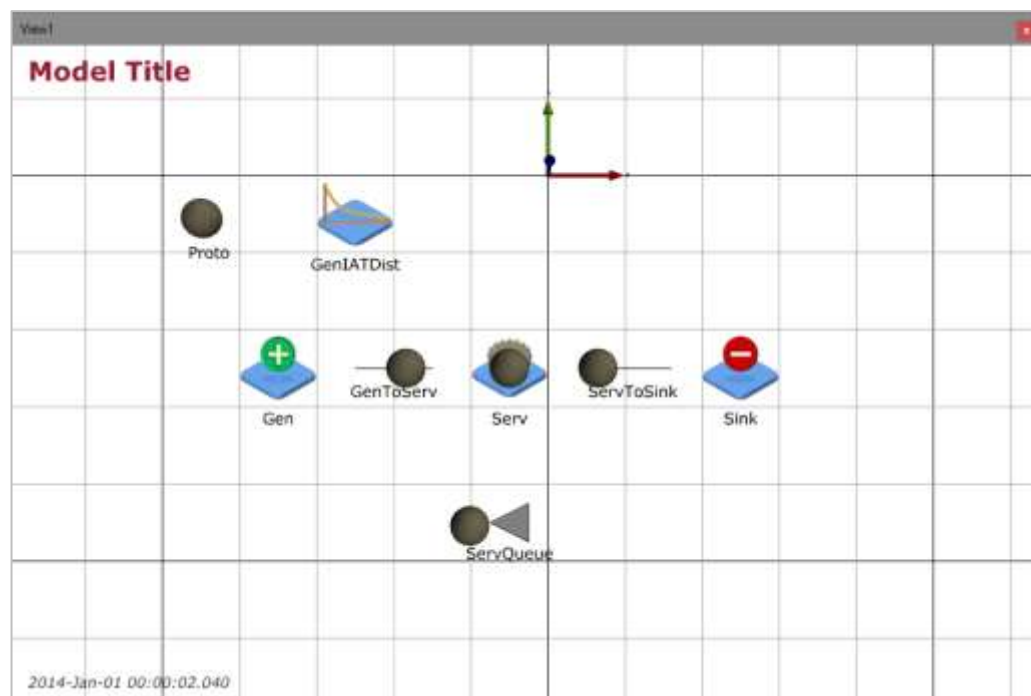
Update the Gen object to sample the GenIATDist distribution for its interarrival time:

Table 3-8 Gen Inputs

| Tab | Keyword | Value |
|------------|------------------|------------|
| Key Inputs | InterArrivalTime | GenIATDist |

Save the model again and press Play. Observe that the Proto entities are now generated randomly, and that there are now times when Proto entities are waiting in ServQueue to be processed, as seen in the following figure.

Figure 3-8 Screenshot of Step 3



3.4 Step 4: Changing Model Graphics

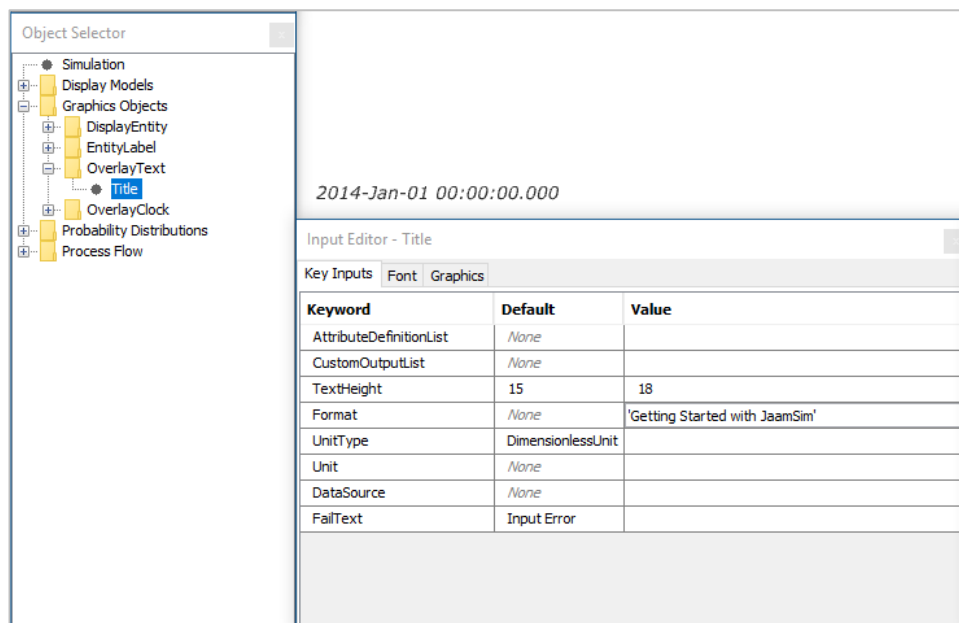
In this step, some graphical adjustments will be made to improve the appearance of the model. While the changes in this step will not impact the functionality of the model, graphics are invaluable when confirming proper operation in more complex models. For the basic example model, the changes are very simple:

- Turn off the XYZ-Axis display
- Turn off the background grid
- Add a title for the model

The first two tasks can be performed by clicking on the Options menu item on the Control Panel and de-selecting the entries 'Show Axes' and 'Show Grid'.

The model's title changed as follows. In the Object Selector, expand the OverlayText palette under Graphics Objects. Now, select the object named 'Title', and enter the new name in the Input Editor under the Format keyword, as shown in Figure 3-9.

Figure 3-9 Changing the Model Title



The model should now look the same as Figure 3-1, shown at the start of this section.

4 Graphical User Interface

The graphical user interface (GUI) consists of the Control Panel, one or more View windows, the Model Builder, the Object Selector, the Input Editor, and the Output Viewer.

4.1 Control Panel

The Control Panel provides a number of run controls and output displays to monitor and control the progress of a simulation run. The Control Panel for an example simulation run is shown in the figure below.

Figure 4-1 JaamSim Control Panel



The Control Panel is divided into two rows, consisting of the Menu Bar and the Tool Bar:

4.1.1 Menu Bar

File

The File entry displays a menu with actions related to saving and loading model input configuration files.

Table 4-1 File Menu

| Menu Entry | Description |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| New | Launches a new-blank model with no objects defined |
| Open | Loads a saved input configuration file |
| Save | Saves the model under the present input configuration file name. |
| Save As... | Saves the current model as a new input configuration file. |
| Import... | Imports one or more 3D models or images. Creates both the ColladaModels/ImageModels containing the graphics and the corresponding DisplayEntities. |
| Print Input Report | Prints the present inputs in a standard file format (.inp). |
| Exit | Closes all windows and exits JaamSim. |

Tools

The Tools entry displays a menu that provides options for showing the windows of the JaamSim graphical user interface:

Table 4-2 Tools Menu

| Menu Entry | Description |
|------------------|---------------------------------------------------------------------------------------------|
| Show Basic Tools | Shows the four main tools: Model Builder, Object Selector, Input Editor, and Output Viewer. |
| Close All Tools | Closes all of the tools that are open. |
| Model Builder | Drag and drop creation and placement of simulation objects. |
| Object Selector | Tree listing of all objects in the present simulation model. |
| Input Editor | View and edit keyword inputs for the selected simulation object. |
| Output Viewer | Key output values of the selected object. |
| Property Viewer | Detailed list of the internal properties of the selected object. |
| Log Viewer | Console for viewing input warnings and error messages. |

The Property Viewer is typically used by programmers who are developing and debugging JaamSim applications, and so its usage is beyond the scope of this manual.

Views

The Views entry displays a menu containing a list of currently defined View windows and provides the ability to create new Views. A View window shows a graphical 3D representation of the model.

Table 4-3 Views Menu

| Menu Entry | Description |
|-----------------|---------------------------------------------------------------------------------------------------|
| View1 | Opens the window for View1, the default View. Does nothing if the window has already been opened. |
| Define New View | Creates a new View object and displays its window. |

Options

The Options entry in the menu bar contains the following entries:

Table 4-4 Options Menu

| Menu Entry | Description |
|--------------------|-------------------------------------------------------------------------------------------------------------------|
| Snap to Grid | If checked, an object being dragged with the mouse will automatically position itself to the nearest grid point. |
| Show Axes | If checked, coordinate axes are displayed at the origin (0, 0, 0) of the coordinate grid. |
| Show Grid | If checked, the coordinate grid is displayed on the xy-plane. |
| Always On Top | If checked, the control panel will always remain on top of other windows. |
| Graphic Debug Info | If checked, information on video memory usage and rendering time will be shown as an overlay on the View windows. |

Units

The Units entry in the menu bar is used to set the units in which to display model outputs in the Control Panel, Output Viewer, and output reports. The default values in the Input Editor will also be displayed in the selected unit.

For example, if you want simulation time to be displayed in seconds instead of hours, select TimeUnit from the submenu of unit types and click on the entry labelled 's'. With this change, the Control Panel will display simulation time in seconds and every output in the Output Viewer with units of time will be displayed in seconds.

Note that the choice of output unit has no effect on the internal calculations in a simulation model. Furthermore, model inputs can always be entered in any valid unit.

Help

The Help entry in the menu bar displays a single menu option that shows the software version number and copyright information.

4.1.2 Tool Bar

The left side of the Tool Bar contains controls for manipulating the simulation run and the 3D view for the active View window. The right side shows the status of the simulation run with parameters such as the elapsed simulation time.

Run Controls

The following controls are provided for starting, pausing, and resetting a simulation model and for controlling its execution speed.




Table 4-5 Run Controls

| Tool Bar Item | Description |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Run/Pause Simulation | Starts, pauses, and resumes the simulation run. |
| Reset Simulation | Stops the model, clears any generated entities, and sets the simulation time to zero. |
| Real Time Mode | If pressed, the simulation speed is held to a constant multiple of wall-clock time. |
| Speed Multiplier | The ratio of simulated time to wall-clock time that is used when the Real Time mode is set. |
| Pause Time: | The time at which the software automatically pauses the simulation. Accepts numbers with time units (e.g. 500 h, 1 y) or date/time format (hh:mm:ss.s or 'YYYY-MM-DD hh:mm:ss.s'). |

Misc. Buttons

A number of buttons are provided to change display features and to assist in model building.

Table 4-6 Misc. Buttons

| Tool Bar Item | Description |
|---------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | Moves the camera for the active View window to a bird's eye view directly above the xy-plane of the simulation and locks it in this position. |
|  Show Entity Flow | When selected, arrows are shown between objects to indicate the flow of entities. |
|  Create Entity Links | When selected, entities are linked when selection is changed. For example, left-clicking on a Server and then on an Assign object, will set the NextComponent input for the Server to the Assign object. |

Run Status

The right side of the Tool Bar consists of indicators to illustrate the progress and status of the simulation run:

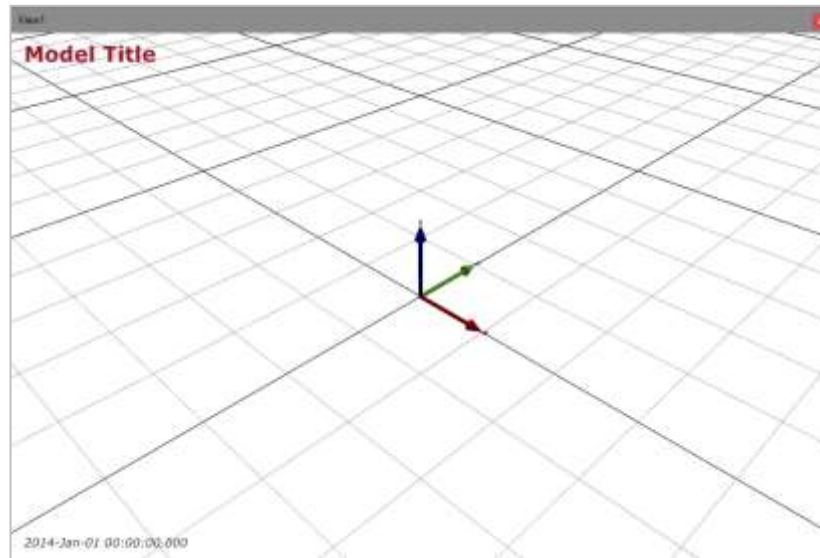
Table 4-7 Run Status

| Tool Bar Item | Description |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Simulation Time | The present simulation time displayed in hours. Other units of time can be selected from the Units entry in the Menu bar (see Section 4.1.1). |
| Run Progress | Percentage of the simulation run that has been completed. |
| Remaining Time | Wall-clock time remaining until the simulation run is completed. |
| Achieved Speed Multiplier | The ratio of elapsed simulated time to wall-clock time. |
| Position | Location of the mouse cursor in the active View window expressed in (x, y, z) coordinates. |

4.2 View Windows

View windows display a graphical representation of a simulation. Multiple View windows can be defined depicting different parts of a model. Each View window is an instance of a View object and can be modified. The following screenshot shows the default View window.

Figure 4-2 Default View Window



4.2.1 Default Graphical Objects

When a new model is created, a default View window is created with some default graphical objects: XY-Grid, XYZ-Axis, Title, and Clock.

The XY-Grid and XYZ-Axis objects are intended as visual aids for placing new objects, and as such they are regular static graphics DisplayEntity objects. The Movable keyword is set to FALSE for both objects, so that they cannot be moved accidentally and do not respond to mouse clicks.

The other two objects are a placeholder for the model title and a clock to show simulation time. These objects are overlay objects (OverlayText and OverlayClock, respectively) that appear in a fixed position on the View window and are not part of the 3D scene.

The position and format of the default objects can be modified through the Input Editor. The Title and Clock objects can be deleted using the Object Selector. The XY-Grid and the XYZ-Axis objects can be turned on or off through Options items in the Menu Bar.

4.2.2 Camera Movement

The basic camera movements are zoom, pan, and orbit. Scrolling the mouse wheel zooms the camera in and out. Clicking and dragging the mouse cursor pans the camera around the View window. Dragging the mouse with the right-button depressed causes the camera to orbit around the current point of interest. These movements are described in more detail in the following table, along with various other useful camera manipulations.

Table 4-8 View Camera Controls

| Mouse/Keyboard Action | Effect |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Left Click | Selects the point of interest. The point on the surface of the object under the cursor becomes the point of interest. If no object is under the cursor, the point on the xy-plane is used. |
| Scroll Wheel | Zooms the camera in or out. The camera is moved towards or away from the point of interest. One click moves the camera 10% closer to or farther away from the point of interest. |
| Left Drag | Pans the xy-plane. The camera is moved in the xy-plane from its present position but the cursor stays fixed on the same point in the View. The point of interest is reset to the cursor position. |
| Shift + Left Drag | Pans the z-axis. The camera is moved along the z-axis from its present position so that the cursor stays fixed on the same point in the View. The point of interest is reset to the cursor position. |
| Right Drag | Orbits the camera. The camera orbits left/right and up/down around the point of interest, following the mouse movement. |
| Shift + Right Drag | Look around. The camera looks left/right and up/down, following the mouse movement. The point of interest is unchanged. |

4.2.3 Moving and Resizing Objects

Individual objects can be moved around using mouse controls that are analogous to the camera controls. To avoid moving an object accidentally, it is necessary to hold the Control key during any movement. After selecting an object in the Object Selector or by clicking on it in a View window, its position, size, and orientation can be manipulated interactively by holding the Control key and dragging the entire object, a corner of the object, or its rotation handle using the mouse. By default, dragging an object moves it in the xy-plane. An object can be moved in the z-direction by holding down both the Control and Shift keys and dragging the object up and down.

These actions are described in more detail in the following table.

Table 4-9 Moving and Resizing Objects

| Mouse/Keyboard Action | Effect |
|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Left Click | Selects the object. The object under the cursor is selected for input/output viewing and for repositioning, resizing, or rotating. The selected object is indicated by a green rectangle bordering it. Green handles allow the object to be resized or rotated. |
| Control + Left Drag | Moves the object parallel to the xy-plane, while holding its z-coordinate constant, following the cursor. |
| Shift + Control + Left Drag | Moves the object parallel to the z-axis, while holding its x-coordinate and y-coordinate constant, following the cursor. |
| Control + Left Drag on a Handle | Resizes/rotates the object. The selected object is resized or rotated using the selected handle. |

4.2.4 Moving and Reshaping Linear Objects

In addition to the standard controls described above, linear objects, such as Arrow objects, can be reshaped by adding, removing, and moving individual points.

The actions for linear objects are described in the following table:

Table 4-10 Moving and Reshaping Linear Objects

| Mouse/Keyboard Action | Effect |
|-------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Left Click on the Line | Selects the line. The line under the cursor is selected for input/output viewing and for repositioning, resizing, or rotating. The selected line turns green, and coloured handles appear at each point in the line. The start of the line is coloured blue, the end is coloured yellow, and intermediate points are coloured green. |
| Control + Left Drag on the Line | Moves the entire line in the xy-plane. The selected line is moved in a plane parallel to the xy-plane, following the cursor. |
| Shift + Control + Left Drag on the Line | Moves the entire line along the z-axis. The selected line is moved along the z-axis, following the cursor. |
| Control + Left Drag on a Handle | Moves location of the point in the xy-plane. The lines on either side of the point adjust to following the point. |
| Shift + Control + Left Drag on a Handle | Moves the location of the point along the z-axis. |
| Alt + Control + Left Click on the Line | Adds a new point and handle to the line. |
| Shift + Alt + Control + Left Click on an Intermediate Point | Deletes the point and reconnects the points on either side of the deleted point. |

4.2.5 Context Menu

Right-clicking displays a list of all objects under the cursor. Selecting one of these objects displays the context menu for the selected object shown in the following table. If only one object is under the cursor, the context menu is displayed right away. A right click is ignored if no object is under the cursor.

Table 4-11 Context Menu Entries

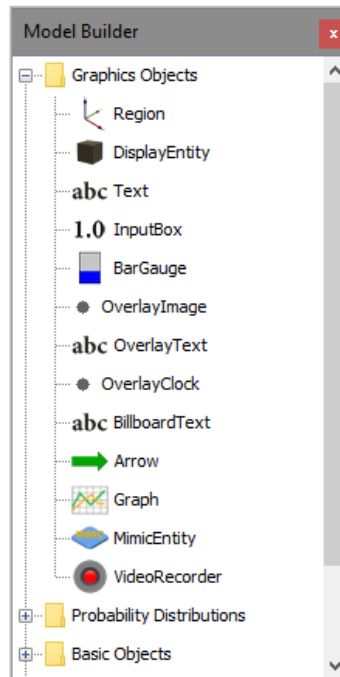
| Menu Item | Description |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Input Editor | Selects the object and opens its Input Editor window. Information for the selected object is shown in Input Editor, Output Viewer, and the Property Viewer. |
| Output Viewer | Selects the object and opens its Output Viewer window. Information for the selected object is shown in Input Editor, Output Viewer, and the Property Viewer. |
| Property Viewer | Selects the object and opens its Property Viewer window. Information for the selected object is shown in Input Editor, Output Viewer, and the Property Viewer. |
| Duplicate | Creates a copy of the selected object in the current View window. |
| Delete | Deletes the selected object. |
| Change Graphics | Opens a dialog box to select a new DisplayModel graphical representation for the selected object. |
| Show Label | Creates an EntityLabel object that displays the present name of the selected object. The object's name can be changed by double-clicking on the EntityLabel and editing the displayed name. |
| Set RelativeEntity | Allows the position of the object to be set relative to the position of a second object. If the second object is moved, the first object moves with it. |
| Set Region | Allows the position of the object to be set in a local coordinate system defined by the Region. If the region is moved or rotated, the objects in the Region move with it. |
| Center in View | Centers the current View on the object. |

4.3 Model Builder

The Model Builder provides palettes of objects that can be dragged and dropped to construct a new model or to modify an existing one.

The following figure shows the Model Builder with the Graphics Objects palette expanded.

Figure 4-3 Model Builder



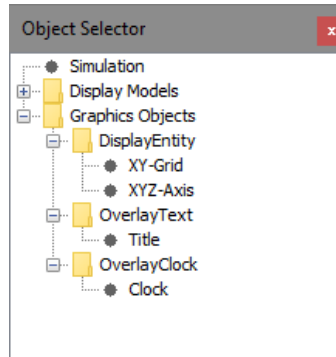
Some objects, such as DisplayModels and Units, do not have a graphical representation and cannot be dragged and dropped by the user. These objects appear in the Object Selector, but not in the Model Builder.

4.4 Object Selector

The Object Selector contains all objects that have been created for the present model, including ones that were created automatically by JaamSim. Objects are grouped according to their palette and type in a tree format that mirrors the structure of the Model Builder. A specific object can be selected either by clicking its node in the Object Selector or by clicking it in a View window.

The following figure shows the Object Selector with the Graphics Objects nodes expanded.

Figure 4-4 Object Selector



Objects that have been created using the Model Builder can be renamed or deleted using the Object Selector. Once an object has been selected, it can be renamed by pressing F2 or by left-clicking on its highlighted entry in the Object Selector, similar to the convention in Windows. A selected object can be deleted by pressing the Delete key, or by right-clicking in either a View window or the Object Selector and selecting Delete.

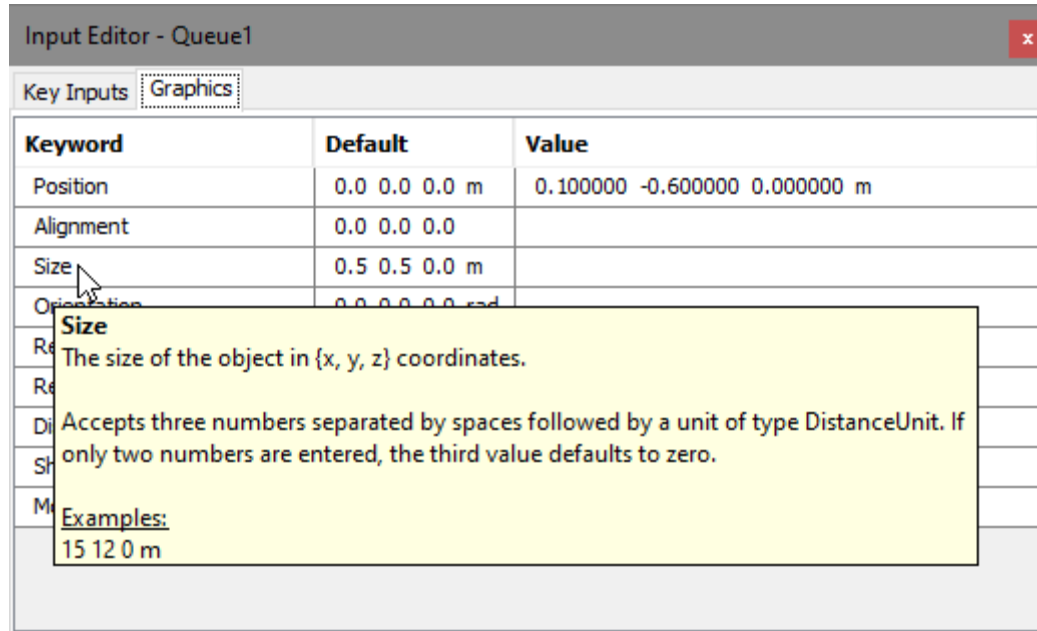
The default objects: XY-Grid, XYZ-Axis, Title, and Clock, can be deleted by the user. However, for the XY-Grid and XYZ-Axis, it is better to hide them by de-selecting their entries in the Options menu, which allows them to be re-instated at a later time.

4.5 Input Editor

The Input Editor allows the user to modify inputs for existing objects or assign inputs to new objects. When an object is selected, its parameters appear in the Input Editor window, grouped under a number of tabs. If a keyword has a default value assigned, it is shown in the Default column.

Hovering the cursor over a keyword will display a tooltip containing a brief description of the keyword and an example input. The Input Editor with the tooltip for Size is shown below.

Figure 4-5 Input Editor



The input for a keyword can be modified by clicking on the entry in the Value column and entering a new value with the appropriate units. Numbers must be entered without spaces or commas and Boolean keywords must take the value TRUE or FALSE (case sensitive). If an entry is made in the Value column, it will overwrite the default. If an input is not valid, an error message will be displayed showing the cause of the error.

Depending on the object, different keyword values will have different data types as shown in the following table.

Table 4-12 Input Data Types

| Data Type | Description | Examples |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------|
| Numbers without units | A number with or without a decimal point. | 5 5.0 |
| Numbers with units | A number followed by a unit separated by one or more spaces. The unit can be enclosed by square brackets, if preferred. | 1000 mm 1.0 m 0.001 km 0.001 [km] |
| Times | Times can be entered normally as a number and time unit or it can be entered in date/time format. The following formats are supported: hh:mm:ss.s, 'YYYY-MM-DD hh:mm:ss.s' YYYY-MM-DDThh:mm:ss.s. Midnight on January 1 of year 0 (0000-01-01) is taken to be zero simulation time regardless of the year displayed by the OverlayClock. If the date/time format includes a space, the entire text must be enclosed by a pair of single quotes. | 30.4 h 30:24:00.0 '0000-01-02 06:24:00.0' 0000-01-02T06:24:00.0 |
| Vectors and Points | Values for the three components followed by a unit. One or more spaces separate the values and unit. If only two values are entered, the z-component is assumed to be zero. The unit can be enclosed by square brackets, if preferred. | 2.0 1.0 0.0 m 2.0 1.0 m 2.0 1.0 0.0 [m] |
| Expressions | A formula containing object outputs and/or mathematical functions. Expressions are described in detail in Section 6.1. | '1 + 2*[Queue1].QueueLength' '2[s] + [Queue1].QueueTimes(1)' |
| Booleans | A value of either TRUE or FALSE (case-sensitive). | TRUE FALSE |
| Colours | A colour can be specified by a colour keyword or by RGB values. A list of named colours and their RGB equivalents are given in Section 0. Transparency can be specified by adding a fourth number after the RGB values. | pink 255 192 203 255 192 203 125 |
| Strings | Text enclosed by a pair of single quotes. The single quotes can be omitted if the text does not include any spaces. | 'Quick red fox' Quick_red_fox |
| Objects | Specified by the object's name. | Server1 Queue1 |

Braces are used to delineate distinct entries in a list. For example, a list of two points would be entered as { 0.0 1.0 0.0 m } { 1.0 1.0 0.0 m }. When an input has only one set of inner braces, it can be entered without the inner braces. For instance, { 1 2 3 } can be entered as 1 2 3.

A drop-down menu is available for many types of inputs. For Boolean inputs, the drop-down menu offers the choice of TRUE or FALSE. For an object input, the drop-down menu lists all the objects of the appropriate type. For a colour input, a suitable colour selector dialog box is provided.

4.6 Output Viewer

The Output Viewer displays the available outputs for the selected object. The values in the Output Viewer are updated continuously as the simulation progresses.

Numerical outputs are normally shown in the appropriate SI unit, except for time which is shown in hours. When it is more convenient to view an output in another unit, the user can specify a set of preferred units using the Units entry in the Menu bar (see Section 4.1.1).

Hovering the cursor over an output name will display a tooltip containing a brief description of the output. The following figure shows the Output Viewer for the case of a Server being selected.

Figure 4-6 Output Viewer

| Output | Value |
|--------------------|-------------|
| Commitment | 0.518304 |
| Availability | 1.00000 |
| Reliability | 1.00000 |
| LinkDevice | |
| obj | Gen_1516257 |
| NumberAdded | 1516257 |
| NumberProcessed | 1516256 |
| NumberInProcessing | |
| ReleaseTime | |
| LinkService | |
| MatchValue | NaN |
| Server | |
| ServiceDuration | 1.00000 s |
| ServicePerformed | 0.794059 s |
| FractionCompleted | 0.794059 |

5 Units

5.1 Unit Types

JaamSim performs all internal calculations in SI units (meters, kilograms, seconds, etc.). However, as it can sometimes be more convenient to specify quantities using other unit systems, JaamSim natively supports the units shown in the following table.

Table 5-1 Supported Unit Types and Units

| Unit Type | Default Unit | Supported Units |
|-------------------------|-----------------------------------------------|----------------------------------------------------------------------|
| DimensionlessUnit | not applicable | not applicable |
| TimeUnit | seconds (s) | ns, us, ms, s, min, h, d, w, y |
| DistanceUnit | meters (m) | mm, cm, m, km, nmi, in, ft, mi |
| SpeedUnit | meters per second (m/s) | m/s, km/h, knots, mph |
| AccelerationUnit | meters per squared second (m/s ²) | m/s ² , ft/s ² |
| MassUnit | kilograms (kg) | kg, t, kt, Mt |
| MassFlowUnit | kilograms per second (kg/s) | (any mass unit)/(s, h, d, y) |
| VolumeUnit | cubic meters (m ³) | m ³ , km ³ , bbl, mbbbl, mmbbl |
| VolumeFlowUnit | cubic meters per second (m ³ /s) | (any volume unit)/(s, h, d, y) |
| AngleUnit | radians (rad) | rad, deg |
| AngularSpeedUnit | radians per second (rad/s) | rad/s, rad/h, deg/s, deg/h |
| EnergyUnit | joules (J) | J, kWh |
| EnergyDensityUnit | joules per cubic meter (J/m ³) | J/m ³ , kWh/m ³ |
| SpecificEnergyUnit | joules per kilogram (J/kg) | J/kg, kWh/t |
| PowerUnit | watts (W) | W, kW, MW |
| CostUnit | dollars (\$) | \$ |
| CostRateUnit | dollars per second (\$/s) | \$/s, \$/h, \$/d |
| LinearDensityUnit | kg/m | kg/m, t/m, kt/m |
| LinearDensityVolumeUnit | m ³ /m | m ³ /m |
| DensityUnit | kg/m ³ | kg/m ³ |
| PressureUnit | Pa | Pa, kPa, psi |
| ViscosityUnit | Pa-s | Pa-s, P, cP |
| AreaUnit | m ² | m ² , cm ² , mm ² , in ² |
| RateUnit | /s | /ns, /us, /ms, /s, /min, /h, /d, /w, /y |

Units are mandatory for most numerical inputs with the exception of pure numbers and ratios. Inputs that are pure numbers are indicated by the DimensionlessUnit type.

5.2 Defining a New Unit

In addition to the predefined units in JaamSim, new units can be specified using a Unit object. A new Unit object can be created by entering the appropriate Define statement in the configuration file (see Section 8). For example if a new TimeUnit called 'Fortnight' is required, then the define statement would be:

```
Define TimeUnit { Fortnight }
```

All Unit objects have the same input keyword:

Table 5-2 Unit Key Inputs

| Keyword | Description |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| ConversionFactorToSI | Two numbers that specify the numerator and denominator, respectively, of the multiplicative factor to convert from the new Unit to SI base units. |

For example, for the 'Fortnight' time unit (two weeks) defined above, the ConversionFactorToSI keyword should be set as follows:

```
Fortnight ConversionFactorToSI ( 1209600 1 )
```

Once the new unit is defined in this way, it can be used with any input that requires that type of unit.

6 Expressions and User-Defined Variables

Most real-world models involve detailed rules and variables that are specific to that application. The variety and complexity of these rules make it impossible to accommodate every possibility in the keywords for a finite set of built-in objects. The expressions, attributes and custom outputs described in this section provide the model builder with the tools necessary to customize the built-in objects for his/her application.

6.1 Expressions

Expressions are mathematical statements that are evaluated by JaamSim. Many keywords that expect a number, string, or entity can also accept an expression that returns the appropriate type of object.

6.1.1 Syntax

Expressions can manipulate and return the following types of entries:

- numbers (with or without units)
- strings
- objects
- arrays
- maps
- lambda functions

Arrays and maps are described in Section 6.1.6. Lambda functions and higher-order functions are described in Section 6.1.7.

The rules for entering numbers, strings, and entities in an expression are given in the following table.

Table 6-1 Numbers, Strings, and Objects

| Entry | Rule | Examples |
|--------|------------------------------------------------------------------------------|-----------------|
| Number | If the number includes a unit, the unit must be enclosed by square brackets. | 1.0 1 [m] |
| String | Strings are enclosed by double quotes. | "Quick red fox" |
| Object | Object names are enclosed by square brackets. | [Server1] |

Expressions are entered as a single line of text using the following syntax.

Table 6-2 Basic Syntax for Expressions

| Rule | Example | Result |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------|
| If an expression includes spaces, curly brackets, or double quotes, it must be enclosed by a pair of single quotes. Spaces in an expression are ignored. | <code>1 [m] + 2 [m]</code> <code>'1 [m] + 2 [m]'</code> <code>"abc"</code> | <code>3 [m]</code> <code>3 [m]</code> <code>abc</code> |
| Rules for mathematical order of operation are respected for the standard operators: +, -, *, /, and ^. Round brackets can be used to modify the order of operation. | <code>1 + 2 * 3</code> <code>(1 + 2) * 3</code> <code>2 * 3 ^ 2</code> <code>(2 * 3) ^ 2</code> | <code>7</code> <code>9</code> <code>18</code> <code>36</code> |
| All calculations must respect units. Unit conversions are performed automatically. | <code>1 [m] + 2</code> <code>1 [m] + 2 [m]</code> <code>1 [m] / 2 [s]</code> <code>1 [m] / 2 [m]</code> <code>1 [m] * 2 [m]</code> | <code>syntax error</code> <code>3 [m]</code> <code>0.5 [m/s]</code> <code>0.5</code> <code>2 [m2]</code> |
| Outputs and attributes are referenced using a dot notation. | <code>[Queue1].QueueLength</code> | Number of entities in Queue1 |
| The reserved string 'this' is used to refer to the object evaluating the expression. | <code>this.State</code> | State of the present entity. |
| Outputs can be chained using the dot notation. | <code>[Queue1].QueueList(1).State</code> | State of the entity at the front of the queue. |

When developing a complex expression it is usually best to prepare it separately in a text editor and copy it to the desired input. Expressions that return a number can be tested using an ExpressionEntity to ensure that correct value is returned.

6.1.2 Local Variables

To improve readability and to avoid repeated calculations, it is possible to define one or more local variables within an expression. A local variable can take the value of any valid type, i.e. a number, string, object, array, map, or lambda function.

The following syntax is used to define a local variable:

```
'<variable1> = <expression1>; ... <variableN> = <expressionN>; <final expression>'
```

For example, the expression `'x = 1; y = 2; x + y'` would return the value 3.

6.1.3 Functions

The following mathematical functions can be used in expressions:

Table 6-3 Basic Mathematical and Logical Functions

| Function | Description | Example | Result |
|------------|---------------------------------------------------------------------------------------------------------------------|--------------------------------|---------------|
| PI | Mathematical constant 'pi' | PI () | 3.14159... |
| E | Mathematical constant 'e' | E () | 2.71828... |
| min | Smallest of a list of values | 'min(1[s], 2[s])' | 1[s] |
| max | Largest of a list of values | 'max(1[s], 2[s])' | 2[s] |
| indexOfMin | Position of the minimum in a list of values | 'indexOfMin(1[s], 2[s])' | 1 |
| indexOfMax | Position of the maximum in a list of values | 'indexOfMax(1[s], 2[s])' | 2 |
| abs | Absolute value | 'abs(-1[s])' | 1[s] |
| ceil | Smallest (closest to negative infinity) integer that is greater than or equal to the argument | 'ceil(5.2[s])' | 6[s] |
| floor | Largest (closest to positive infinity) integer that is less than or equal to the argument | 'abs(5.2[s])' | 5[s] |
| signum | Zero if the argument is zero, 1.0 if the argument is greater than zero, and -1.0 if the argument is less than zero. | 'signum(5.2[s])' | 1 |
| sqrt | Square root | 'sqrt(4.0)' | 2.0 |
| cbrt | Cube root. | 'cbrt(8.0)' | 2.0 |
| % | Modulus (remainder) operator. Used as an operator, not a function. | '11.5 % 4' '11.5[s] % 4[s]' | 3.5 3.5[s] |
| choose | Selects from a list using an index | 'choose(2, 1[s], 2[s])' | 2[s] |

Table 6-4 Exponential and Trigonometric Functions

| Function | Description | Example | Result |
|----------|--------------------------------------------------------------------------------------------------------------------------------------------|--------------------|------------|
| exp | Exponential function | 'exp(1)' | 2.71828... |
| ln | Natural logarithm | 'ln(2.71828)' | 0.999999 |
| log | Base-10 logarithm | 'log(100)' | 2.0 |
| sin | Sine of an angle or dimensionless number | 'sin(30[deg])' | 0.5 |
| cos | Cosine of an angle or dimensionless number | 'cos(60[deg])' | 0.5 |
| tan | Tangent of an angle or dimensionless number | 'tan(45[deg])' | 1 |
| asin | Arcsine function | 'asin(0.5)' | 30[deg] |
| acos | Arccosine function | 'acos(0.5)' | 60[deg] |
| atan | Arctangent function | 'atan(1.0)' | 45[deg] |
| atan2 | Two-argument arctangent function. For Cartesian coordinates x and y, atan2(x,y) returns the angle for the corresponding polar coordinates. | 'atan2(1.0, -1.0)' | 135[deg] |

Table 6-5 Functions Involving Objects

| Function | Description | Example | Result |
|----------|---------------------------------------------------------------------------------------------------------|---------------------|--------|
| notNull | Determines whether an entity exists. It can be used to test whether an output such as obj has been set. | 'notNull(this.obj)' | 0 or 1 |

Table 6-6 Functions Involving Strings

| Function | Description | Example | Result |
|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------|
| + | Strings can be concatenated using the + operator. If necessary, the right hand side of the + operator is converted to a string. | '"abc" + "def" ' '"abc" + 5[m] ' '"abc" + [Entity1] ' | "abcdef" "abc5.0[m]" "abc[Entity1]" |
| format | Constructs a string using a format string and one or more additional arguments. Each argument can be a dimensionless number, string, object, array, or map. The function mirrors the <code>String.format</code> method provided in Java. Each entry of % in the format string indicates a format code that inserts the next argument in the resulting output text. The full set of Java format codes is supported. The two most relevant are: %s - displays any argument as a string. %.nf - displays a number with n decimal places. | 'format("x = %s",[Entity1]) ' 'format("x = %s",5) ' 'format("x = %.3f",5) ' 'format("x = %s cm", 5[m]/1[cm]) ' 'format("%s x %s","abc","def") ' | "x = [Entity1]" "x = 5.0" "x = 5.000" "x = 500.0 cm" "abc x def" |

6.1.4 Logical Operations

Logical operations can also be performed in Expressions. All non-zero dimensionless values are interpreted as `TRUE`, while zero is interpreted as `FALSE`. The following are examples of syntactically valid logical Expressions:

```
'this.OutputA >= 1'
'(this.OutputA >= 1) && ([Entity1].OutputB == 0)'
```

The following logical operators are supported:

Table 6-7 Logical Operators

| Operator | Description | Example | Result |
|----------|--------------------------|--------------|--------|
| == | Equal to | '4.2 == 4.2' | 1 |
| != | Not equal to | '3.5 != 4.2' | 1 |
| < | Less than | '3.5 < 4.2' | 1 |
| <= | Less than or equal to | '3.5 <= 3.5' | 1 |
| > | Greater than | '4.2 > 3.5' | 1 |
| >= | Greater than or equal to | '3.5 >= 3.5' | 1 |
| && | Logical AND operation | '1 && 1' | 1 |
| | Logical OR operation | '1 0' | 1 |
| ! | Logical NOT operation | '! 0' | 1 |

The && and || operators use short-circuited evaluation, that is, the right-hand side of the operator is only evaluated when necessary. For example, if the left-hand side of the && operator is FALSE, then the result is FALSE regardless of the value of the right-hand side. Similarly, if the left-hand side of the || operator is TRUE, then the result is TRUE regardless of the right-hand side.

6.1.5 Conditionals

Conditional operations are implemented with the ternary operator, `?`, using the following syntax:

```
<condition_expression> ? <true_expression> : <false_expression>
```

The `?` operator uses short-circuited evaluation, that is, the `<true_expression>` is evaluated only when the `<condition_expression>` is TRUE and the `<false_expression>` is evaluated only when it is FALSE.

Examples of the conditional operator are:

```
'2>1 ? 5 : 4'
'this.OutputA >= 2 ? [Entity1].OutputB + 1 : 0'
```

Complex logical expressions can be constructed by chaining a series of ternary operators to form an “if, else-if” type structure, e.g.:

```
'this.OutputA <= 2 ? 2 : (this.OutputA <= 4 ? 4 : 6)'
```

Note that brackets were added to this expression to improve readability – they are not mandatory.

6.1.6 Arrays and Maps

An array is a sequence of entries that are indexed by an integer value starting with 1. A map is similar to an array except that its entries are indexed by a key, such as a string, instead of an integer. The entries in an array or map can be numbers, strings, entities, or other arrays or maps.

Table 6-8 Arrays and Maps

| Entry | Rule | Examples |
|-------|----------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------|
| Array | Arrays are enclosed by curly braces, with individual entries separated by commas. | <pre>{ 5, 6, 7 } { 5[m], 6[m], 7[m] } { "a", "b", "c" } { [Queue1], [Queue2] } { {1, 2}, {3, 4} }</pre> |
| Map | The output "StateTimes" is the only map object used in JaamSim at present. Users cannot create new maps. | <code>[Server1].StateTimes</code> |

Arrays and maps can be referenced in an expression using the following syntax.

Table 6-9 Syntax for Arrays and Maps

| Rule | Example | Result |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------|------------------------------------------------|
| Entries in an array are referenced by specifying an index. The index value can be either a constant or an expression that returns a dimensionless number. A non-integer value for the index will be truncated. | <pre>[Queue1].QueueList(2) {5,6,7}(2)</pre> | <p>Second entity in the queue.</p> <p>6</p> |
| Entities in a map are referenced by specifying a key (usually a string). | <code>[Server1].StateTimes("Idle")</code> | Total time Server1 has been in its Idle state. |
| Entries in a nested array are referenced by providing multiple indices. | <code>'{ {5,6}, {7,8} }(2)(1) '</code> | 7 |
| Arrays and maps can be concatenated and appended using the + operator. | <pre>'{1,2,3} + {4,5,6} ' '{1,2,3} + "abc" '</pre> | <pre>{1,2,3,4,5,6} {1,2,3,"abc" }</pre> |

A number of functions are provided that take an array or map as an argument.

Table 6-10 Functions of Arrays and Maps

| Function | Description | Example | Result |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------|---------------------------------------|
| size | Number of entries in an array or map. | 'size({5,-1,2})' | 3 |
| minCol | Smallest entry in an array or map. | 'minCol({5,-1,2})' | -1 |
| maxCol | Largest entry in an array or map. | 'maxCol({5,-1,2})' | 5 |
| indexOfMinCol | Index or key of the smallest entry in an array or map. If this value appears in several entries, the index of the first one is returned. | 'indexOfMinCol({5,-1,2})' | 2 |
| indexOfMaxCol | Index or key of the largest entry in an array or map. | 'indexOfMaxCol({5,-1,2})'' | 1 |
| indexOf | Index or key of the specified entry in an array or map. | 'indexOf({5,-1,2}, -1) | 2 |
| indexOfNearest | Index or key of the entry in an array or map that is closest to the specified value. | 'indexOfNearest({5,-1,2}, 1.5)' | 3 |
| range | Generates an array of numerical values that can be used as an input to a higher-order function. | range(3) range(2,4) range(2,3,0.5) range(2,1) | {1,2,3} {2,3,4} {2,2.5,3} {} |

When several entries in an array have the same value, the functions `indexOfMinCol`, `indexOfMaxCol`, `indexOf` and `indexOfNearest` return the first index that satisfies the condition. When several entries in a map have the same value, the index returned by these functions is the first one that was entered in the map.

6.1.7 Lambda Functions and Higher-Order Functions

A lambda function is an expression that takes one or more input variables and returns a number, string, object, array, map, or another lambda function. Lambda functions can be used with higher-order functions to perform complex calculations that would otherwise require a loop structure.

Table 6-11 Lambda Functions

| Entry | Rule | Examples |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| Lambda Function | Input variables are enclosed by bars and separated by commas. The expression that generates the returned value is enclosed by brackets. Input variables can be a number, string, array, map, or another lambda function. The object returned can be any of these same types of objects. | x,y (x + y) |

Table 6-12 Syntax for Lambda Functions

| Rule | Example | Result |
|------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------|-------------------|
| A lambda function can be evaluated by providing input values enclosed by brackets. | $ x (2*x) (5)$ $ x,y (x+2*y) (1,2)$ $ x,y (x+y) ("abc", "def")$ | 10 5 abcdef |
| A lambda function with multiple inputs can be turned into one with fewer inputs by providing one or more of the inputs | $ x,y (x+2*y) (1)$ | $ y (1+2*y)$ |

A higher-order function is one that takes a lambda function as an argument.

Table 6-13 Higher-Order Functions

| Function | Description | Example | Result |
|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------|
| map | Applies a one-input lambda function to each element of an array and returns an array with the resulting values. If a two-input lambda function is specified, the second input is the index of the element in the array. | $\text{map}(x (2*x), \{1,2\})$ $\text{map}(x,i (2*i), \{5,8\})$ | $\{2,4\}$ $\{2,4\}$ |
| filter | Applies a one-input lambda function to each element of an array and returns an array with only the ones that return TRUE (i.e. a non-zero number). If a two-input lambda function is specified, the second input is the index of the element in the array. | $\text{filter}(x (x>2), \{1,2,3,4\})$ $\text{filter}(x,i (i>2), \{5,6,7,8\})$ | $\{3,4\}$ $\{7,8\}$ |
| reduce | Applies the first input of a two-input lambda function to each element of an array. The second input to the reduce function is the initial value for an internal value maintained by the function during the calculation. The result of the calculation for each element is assigned to this internal value. After the last element is processed, the internal value is returned. The reduce function has three inputs: the function, the initial value to be assigned to the internal value, and the array to be processed. | $\text{reduce}(x, \text{accum} (x+\text{accum}), 0, \{1,2,3\})$ $\text{reduce}(x, \text{accum} (\max(x, \text{accum})), 0, \{1,2,3\})$ $\text{reduce}(x, \text{accum} (x \text{accum}), 0, \{0,1,0\})$ | 6 3 1 |
| sort | Applies a two-input lambda function to the elements of an array and returns an array that has been re-ordered so that the lambda function returns TRUE (i.e. a non-zero number) for each adjacent pair of elements. The lambda function should return FALSE for entries that are equal. | $\text{sort}(x,y (x>y), \{2,3,1\})$ | $\{3,2,1\}$ |

6.2 User-Defined Variables

Users can define two types of variables for individual objects in a model:

- **Attribute.** A variable whose value can be changed by one or more Assign objects in a model (see Section 14.12). Attributes are a useful way to add application-specific information to generated entities or to the permanent objects in a model. For example, if there are two types of customers in a model that require different service times, an attribute named "type" can be added to the generated customer objects and randomly assigned the value 1 or 2. When the customer arrives at the server, its service time can then be calculated based on its type attribute.
- **Custom Output.** A variable whose value is calculated on demand by evaluating a specified expression during the simulation run. Custom outputs are a way for the user to supplement the JaamSim's built-in outputs or to avoid repeating a complicated calculation in more than one expression. Note that a custom output cannot use another custom output in its definition.

Attributes and custom outputs can have same types of values as expressions, i.e. numbers with or without units, strings, entities, arrays, maps, or lambda functions. They appear in the Output Viewer as outputs under the Entity heading.

The name for an attribute or custom output can be any alphanumeric text provided that no spaces are included.

Attributes and custom outputs can be defined for any of the objects in a model using the AttributeDefinitionList and CustomOutputList keywords described in the following table.

Table 6-14 Keywords for Attributes and Custom Outputs

| Keyword | Description |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AttributeDefinitionList | <p>Defines one or more attributes for the object. The following format is used for the input:</p> <pre>{ <Name1> <InitValue1> } ... { <NameN> <InitValueN> }</pre> <p>where Name is the name of the attribute and InitValue is an expression that returns the initial value for the attribute. The expression is evaluated when the simulation is first started or re-started. Normally, the expression is a simple constant, such as 5[km]. The attribute type (number, string, etc.) is determined by its initial value.</p> <p>The following example defines a series of attributes of various types:</p> <pre>{ A 1 } { B 9[km] } { C "abc" } { D [Server1] } { E '{5,3,7}' }</pre> |

| Keyword | Description |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CustomOutputList | <p>Defines one or more custom outputs for the object. The following format is used for the input:</p> <pre>{ <Name1> <Exp1> <Unit1> } ... { <NameN> <ExpN> <UnitN> }</pre> <p>where Name is the name of the custom output, Exp is the expression to be evaluated on demand, and Unit is the unit type for the output.</p> <p>For example, the following example defines a custom output called TwiceSimTime, whose value is equal to two times the present simulation time:</p> <pre>{ TwiceSimTime '2 * this.SimTime' TimeUnit }</pre> |

7 Simulation Runs and Experiments

7.1 Simulation Object

The Simulation object is used to store inputs that define basic parameters of the model, such as run duration. The Simulation object is created automatically when a new model is started, and thus does not appear in the Model Builder.

Table 7-1 Simulation Inputs

| Keyword | Description |
|----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| RunDuration | Duration of the simulation run in which statistics will be recorded. |
| InitializationDuration | The initialization interval for the simulation run. The model will run for the InitializationDuration interval and then clear the statistics and execute for the specified RunDuration interval. The total length of the simulation run will be the sum of the InitializationDuration and RunDuration inputs. |
| PauseCondition | An optional expression that pauses the run when TRUE is returned. |
| ExitAtPauseCondition | If TRUE, the simulation run will be terminated when the PauseCondition expression returns TRUE. If multiple runs have been specified, then the next run will be started. If no more runs have been specified, the simulation will be paused or terminated depending on the input to the ExitAtStop keyword. |
| ExitAtStop | If TRUE, the program will be closed on completion of the last simulation run. Otherwise, the last run will be paused. |
| GlobalSubstreamSeed | Global seed that sets the substream for each probability distribution. Must be an integer ≥ 0 . GlobalSubstreamSeed works together with each probability distribution's RandomSeed keyword to determine its random sequence. It allows the user to change all the random sequences in a model with a single input. To run multiple replications, set the appropriate inputs under the Multiple Runs tab and then set the GlobalSubstreamSeed input to the run number or to one of the run indices. |
| PrintReport | If TRUE, a full output report is printed to the file <configuration file name>.rep at the end of the simulation run. |
| ReportDirectory | The directory where output files will be written. The default location is the directory of the input configuration file. |
| UnitTypeList | A list of the unit types for the selected outputs specified by the RunOutputList keyword. Use DimensionlessUnit for a text output. |
| RunOutputList | One or more selected outputs to be printed at the end of each simulation run. Each output is specified by an expression. In script mode (-s tag), the selected outputs are printed to the command line (standard out). Otherwise, they are printed to the file <configuration file name>.dat. |

| | |
|-----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| TickLength | The smallest time increment for JaamSim's internal integer-based time keeping. The default value of 1 microsecond will support simulation runs of more than 100 years. |
| <u>Multiple Runs</u> | |
| RunIndexDefinitionList | Defines the number of run indices and the maximum value N for each index. When making multiple runs, each index will be iterated from 1 to N starting with the last index. One run will be executed for every combination of the run index values. For example, if three run indices are defined with ranges of 3, 5, and 10, then at total of $3*5*10 = 150$ runs will be executed. |
| StartingRunNumber | The first run number to be executed. The value can be entered as either an integer or as the equivalent combination of run indices. For example, if there are three run indices with ranges of 3, 5, and 10, then run number 22 can be expressed as 1-3-2 because $22 = (1-1)*5*10 + (3-1)*10 + 2$. |
| EndingRunNumber | The last run number to be executed. The value can be entered as either an integer or as the equivalent combination of run indices. For example, if there are three run indices with ranges of 3, 5, and 10, then run number 78 can be expressed as 2-3-8 because $78 = (2-1)*5*10 + (3-1)*10 + 8$. |
| <u>GUI</u> | |
| DisplayedUnits | An optional list of units to be used for displaying model outputs. |
| SnapToGrid | If TRUE, a dragged object will be positioned to the nearest grid point. |
| SnapGridSpacing | The distance between snap grid points. |
| IncrementSize | The distance moved by the selected entity when the an arrow key is pressed. |
| RealTime | If TRUE, the simulation is executed a constant multiple of real time. Otherwise, the run is executed as fast as possible, limited only by processor speed. |
| RealTimeFactor | The target ratio of elapsed simulation time to elapsed real time. |
| PauseTime | The time at which the simulation will be paused. |
| ShowModelBuilder | If TRUE, the Model Builder tool is shown on startup. |
| ShowObjectSelector | If TRUE, the Object Selector tool is shown on startup. |
| ShowInputEditor | If TRUE, the Input Editor tool is shown on startup. |
| ShowOutputViewer | If TRUE, the Output Viewer tool is shown on startup. |
| ShowPropertyViewer | If TRUE, the Property Viewer tool is shown on startup. |
| ShowLogViewer | If TRUE, the Log Viewer tool is shown on startup. |

7.2 Performing Multiple Simulation Runs

Multiple simulation runs can be executed automatically, one after another, using the keywords under the Multiple Runs tab for the Simulation object. The Simulation outputs RunNumber and RunIndex are used to change selected inputs between simulation runs. By default, the RunNumber output starts at 1 and is incremented by one with each simulation run that is performed. This output can be used to vary one or more inputs by referencing [Simulation].RunNumber in an expression. For example, setting the ServiceTime input for a Server to the following expression:

```
'1[s] + 0.1[s]*[Simulation].RunNumber'
```

assigns the service time to 1.1 s, 1.2 s, 1.3 s, etc., as the run number is incremented over multiple runs.

The RunIndex output is used when there are multiple inputs to test. This output contains an array of integers that are each incremented from 1 to N, where a separate value for N can be specified for each index. The number of run indices and the ranges over which they are incremented are determined by the RunIndexDefinitionList keyword.

For example, suppose there are two Servers and service times of 1.1 s, 1.2 s, and 1.3 s are to be tested for Server1 and service times of 2.1 s and 2.2 s are to be tested for Server2. Ten replications are to be made for each combination of service times. In this case, three run indices are needed – one for each variable that is to be changed. The run indices are defined by entering the values 3 2 10 to the RunIndexDefinitionList keyword. This input indicates that RunIndex(1) will be incremented over the range 1 to 3, RunIndex(2) will be incremented over 1 to 2, and that RunIndex(3) will be incremented over 1 to 10. The three run indices are used in the model inputs as follows:

- ServiceTime keyword for Server1: '1[s] + 0.1[s]*[Simulation].RunIndex(1)'
- ServiceTime keyword for Server2: '2[s] + 0.1[s]*[Simulation].RunIndex(2)'
- GlobalSubstreamSeed keyword for Simulation: '[Simulation].RunIndex(3)'

With these inputs, a total of sixty runs would be performed with the run indices incremented in the following sequence:

- 1-1-1, 1-1-2, ... 1-1-10,
- 1-2-1, 1-2-2, ... 1-2-10,
- 2-1-1, 2-1-2, ... 2-1-10,
- 2-2-1, 2-2-2, ... 2-2-10,
- 3-1-1, 3-1-2, ... 3-1-10,
- 3-2-1, 3-2-2, ... 3-2-10,

The notation $i-j-k$ indicates run indices RunIndex(1) = i , RunIndex(2) = j , and RunIndex(3) = k .

It is not necessary to perform all the simulation runs defined by the run indices. The Simulation keywords StartingRunNumber and EndingRunNumber can be used to determine the runs that will be performed. To perform all sixty runs, the StartingRunNumber should be set to 1 (the default value) and the EndingRunNumber should be set to 60.

Run indices are related to the run number by a mathematical equation that performs the necessary transformation. In the example given above, the RunNumber increases from 1 to 60, at the same time as the run indices increase from 1-1-1 to 3-2-10. Run numbers and run indices in the $i-j-...-k$

notation can be used interchangeably for the StartingRunNumber and EndingRunNumber inputs. For example, to perform all sixty runs, the StartingRunNumber could be set to 1-1-1 and EndingRunNumber to 3-2-10 instead of 1 and 60.

7.3 Customized Output Report

When making multiple runs, the output reports for the runs are appended one after another in a single file. If more than just a few runs are to be made, it is best to create a customized output report using the RunOutputList keyword for Simulation. This keyword allows the user to specify the set of outputs needed to analyse the results from the runs. Each output is entered as a separate expression enclosed by curly braces. For example, if the desired outputs are the simulation run number, average utilisation of Server1, and average waiting time for Queue1, then the input to the RunOutputList keyword would be as follows:

```
{ [Simulation].RunNumber } { [Server1].Utilisation } { [Queue1].AverageQueueTime }
```

It is best to prepare this input in a text editor and to copy each entry one-by-one to the Input Editor. This method allows errors to be located and corrected more easily.

Before entering the RunOutputList input, it is first necessary to specify the unit type for each output using the UnitTypeList keyword. For this example, the input would be:

```
DimensionlessUnit DimensionlessUnit TimeUnit
```

At the end of each run, the specified outputs are collected in a single output file with the name <configuration file name>.dat. The file has one row for each simulation run that was performed and is tab-delimited, which allows it to be imported directly into Excel for analysis.

The RunOutputList report can be generated with or without the main report, as specified by the PrintReport keyword.

8 Configuration File

The easiest way to create a simple model in JaamSim is to use the Graphical User Interface. However, once a model becomes more complex, it is often easier to edit the configuration file (CFG file) in a text editor. The configuration file is saved in plain text and has been designed to be human readable.

There are many advantages to a readable input file in plain text:

- Inputs can be easily reviewed and audited.
- Standard software for change control such as GIT can be used to track model inputs.
- Software for performing simulation experiments and optimisation can be developed by third-parties in other programming languages such as Python.

The recommended text editor is Notepad++, an open-source editor available for download at www.notepad-plus-plus.org.

8.1 Basic Structure

A JaamSim input configuration file consists of a series of lines, akin to a scripting language. Each line consists of a combination of object names, keywords, and values contained within braces. One or more spaces are used to separate these elements. Braces are also used to denote sets of arguments within the outer braces required for arguments in general. Blank lines are ignored by JaamSim.

Lines beginning with a hash mark ('#') can be used to create comments to document the input files. If a comment extends for several lines, each line must start with a hash mark.

8.2 Object Definitions

In JaamSim, an object is initialized by a Define statement. The statement contains Define followed by the object type, and the object name enclosed by braces. Multiple objects can be defined at the same time, provided that they are of the same type. For instance, the following two lines define respectively a single Arrow object and three Arrow objects.

```
Define Arrow { SingleArrow }
Define Arrow { Arrow1 Arrow2 Arrow3 }
```

Object instances can only be referenced after they have been defined.

8.3 Object Inputs

Once an object is defined, its keyword values can be set using a command of the following form:

```
<object name> <keyword> { <value1> <value2> ... }
```

where value1, value2, ... is the list of values for the keyword separated by one or more spaces. For instance, the following line sets the colour of the Arrow1 object to be black:

```
Arrow1 Colour { black }
```

Multiple parameters for an object can be set in one line containing the object name followed by keyword and value pairs.

```
Arrow1 Colour { black } Width { 2 }
```

Inner braces are used for keywords that accept multiple input values.

```
Arrow1 Points { { 0 0 0 m } { 1 1 1 m } }
```

8.4 Include Statements

The user can store input data in multiple files and then refer to these files in an input configuration file using Include statements. These statements refer to other input configuration files by filename and path, surrounded by single quotes:

```
Include '..\Base File\InputFile.cfg'
```

Include statements are particularly useful when only a few inputs are varied across many simulation runs. Include statements can be used to create incremental configuration files for additional runs that contain a base case configuration file :

```
Include '..\Base File\Basecase.cfg'
Arrow1 Width { 2.0 }
```

This example includes the contents of Basecase.cfg and modifies the already-defined object Arrow1's keyword Width value to 2.0. Note that the changes from the base case configuration must appear after the Include statement. These simple configuration files are useful because it is easy to tell exactly how the configuration differs from the base case configuration.

8.5 Groups

Group objects bundle multiple objects together to simplify inputs. Instead of referring to a long list of objects, a single Group can be used instead. The Group may be used to set the value for a keyword for all members instead of setting the value for each member of the Group. Certain keywords also accept Group objects as values.

Table 8-1 Group Inputs

| Keyword | Description |
|---------|---------------------------------------------------------------------------|
| List | A list of names of the objects included in this list, enclosed by braces. |

The following example demonstrates the use of Groups:

```
Define      Arrow      { Arrow1 Arrow2 Arrow3 }
Define      Group      { ArrowList }
ArrowList   List        { Arrow1 Arrow2 Arrow3 }
ArrowList   Colour      { black }
```

In this example, a Group of three Arrow objects is created and each Arrow is set to the colour black.

By using the List keyword, a fourth Arrow can be added to the Group:

```
Define      Arrow      { Arrow4 }
ArrowList   List        { ArrowList Arrow4 }
ArrowList   Colour      { black }
```

8.6 RecordEdits Statement

The RecordEdits statement is used to preserve the organisation and formatting of a configuration file that has been prepared manually by the user.

It is usually best to construct a complex model manually using a text editor. These inputs are carefully formatted and organised, and include comments to document model design. However, once this material has been created, the easiest way to position the objects and to add graphics such as titles, labels, etc. is through the graphical user interface. If the model is then saved, all the formatting and comments would normally be lost.

JaamSim avoids this predicament with the RecordEdits statement. On saving, JaamSim copies all inputs before the RecordEdits statement line-by-line to the saved file, and then saves all the changes to the model using computer-written inputs. The following example illustrates this structure:

```
" Manually prepared inputs:
" - Everything before the RecordEdits statement is unchanged when JaamSim saves a
file.

RecordEdits

" Computer written inputs:
" - Everything that appears after the RecordEdits statement is written by the
computer.
```

The Save functionality in JaamSim is disabled when an input file is loaded without a RecordEdits statement. In this case, the Save As operation adds a RecordEdits statement to the end of the original configuration file and then writes out the new inputs.

8.7 Example Configuration File

The configuration file for the Basic Example described in Section 3 is given below.

```
RecordEdits

Define ColladaModel { Grid100x100 Axis }
Define DisplayEntity { XY-Grid XYZ-Axis }
Define View { View1 }
Define TextModel { TitleTextModel ClockTextModel }
Define OverlayText { Title }
Define OverlayClock { Clock }
Define SimEntity { Proto }
Define EntityGenerator { Gen }
Define EntityConveyor { GenToServ ServToSink }
Define Server { Serv }
Define EntitySink { Sink }
Define Queue { ServQueue }
Define ExponentialDistribution { GenIATDist }

GenIATDist UnitType { TimeUnit }

Simulation Description { 'Simulation run control inputs' }
Simulation SnapToGrid { TRUE }
Simulation RealTime { TRUE }
Simulation RealTimeFactor { 2048 }
Simulation PauseTime { }
Simulation ShowModelBuilder { TRUE }
Simulation ShowObjectSelector { TRUE }
```



```

Simulation ShowInputEditor { TRUE }
Simulation ShowOutputViewer { TRUE }
Simulation ShowPropertyViewer { FALSE }
Simulation ShowLogViewer { FALSE }

Grid100x100 ColladaFile { <res>/shapes/grid100x100.dae }

XY-Grid Description { 'Grid for the X-Y plane (100 m x 100 m)' }
XY-Grid Size { 100 100 m }
XY-Grid DisplayModel { Grid100x100 }
XY-Grid Movable { FALSE }

Axis ColladaFile { <res>/shapes/axis_text.dae }

XYZ-Axis Description { 'Unit vectors' }
XYZ-Axis Alignment { -0.4393409 -0.4410096 -0.4394292 }
XYZ-Axis Size { 1.125000 1.1568242 1.1266404 m }
XYZ-Axis DisplayModel { Axis }
XYZ-Axis Show { FALSE }
XYZ-Axis Movable { FALSE }

View1 Description { 'Default view window' }
View1 ViewCenter { -0.299610 -2.582932 2.866546 m }
View1 ViewPosition { -0.299610 -2.582933 11.526800 m }
View1 ShowWindow { TRUE }
View1 SkyboxImage { <res>/images/sky_map_2048x1024.jpg }

TitleTextModel Description { 'Text style for the Title' }
TitleTextModel FontColour { 150 23 46 }
TitleTextModel FontStyle { BOLD }

ClockTextModel Description { 'Text style for the Clock' }
ClockTextModel FontColour { 51 51 51 }
ClockTextModel FontStyle { ITALIC }

Title Description { 'Title for the simulation model' }
Title TextHeight { 18 }
Title Format { 'Getting Started with JaamSim' }
Title Position { 0.000000 0.000000 0.000000 m }
Title DisplayModel { TitleTextModel }
Title ScreenPosition { 15 15 }

Clock Description { 'Simulation date and time (no leap years or leap seconds)' }
Clock TextHeight { 10 }
Clock StartingYear { 2014 }
Clock DateFormat { 'yyyy-MMM-dd HH:mm:ss.SSS' }
Clock DisplayModel { ClockTextModel }
Clock ScreenPosition { 15 15 }
Clock AlignBottom { TRUE }

Proto Position { -4.400000 -0.600000 0.000000 m }
Proto Alignment { 0.0 0.0 -0.5 }

Gen NextComponent { GenToServ }
Gen InterArrivalTime { GenIATDist }
Gen PrototypeEntity { Proto }
Gen Position { -3.500000 -2.500000 0.000000 m }

GenToServ NextComponent { Serv }
GenToServ TravelTime { 1 s }
GenToServ Position { -2.500000 -2.500000 0.000000 m }
GenToServ Points { { -2.500 -2.500 0.000 m } { -1.500 -2.500 0.000 m } }

Serv NextComponent { ServToSink }
Serv WaitQueue { ServQueue }

```

```
Serv ServiceTime { 1 s }
Serv Position { -0.500000 -2.500000 0.000000 m }

ServToSink NextComponent { Sink }
ServToSink TravelTime { 1.5 s }
ServToSink Position { 0.600000 -2.500000 0.000000 m }
ServToSink Points { { 0.600 -2.500 0.000 m } { 1.600 -2.500 0.000 m } }

Sink Position { 2.500000 -2.500000 0.000000 m }

ServQueue Position { -0.500000 -4.500000 0.000000 m }

GenIATDist RandomSeed { 1 }
GenIATDist MaxValue { 10 s }
GenIATDist Mean { 2 s }
GenIATDist Position { -2.500000 -0.500000 0.000000 m }
```

9 Maintenance, Breakdowns, and Thresholds

A significant feature of JaamSim is its ability to model planned maintenance, breakdowns, and process blockages, and to record the total time the process spends in each state. This feature allows a JaamSim model to combine a full Reliability, Availability, and Maintainability (RAM) analysis, normally done separately with specialised software, with a typical logistics model.

9.1 Thresholds

Process blockages are modelled by Threshold objects that open and close according to various types of rules:

- **TimeSeriesThreshold** (Section 13.3). Opens and closes when the value provided by a TimeSeries object (Section 13.2) meets various conditions.
- **ExpressionThreshold** (Section 13.4). Opens and closes according to a specified expression that returns TRUE or FALSE.
- **SignalThreshold** (Section 14.19). Opens and closes when an entity is received by an EntitySignal object (Section 14.18).

All types of Threshold objects share the following inputs and outputs.

Table 9-1 Threshold Inputs

| Keyword | Description |
|----------------|-------------------------------------------------------------------------|
| OpenColour | The colour of the threshold graphic when the threshold is open. |
| ClosedColour | The colour of the threshold graphic when the threshold is closed. |
| ShowWhenOpen | A Boolean value. If TRUE, the threshold is displayed when it is open. |
| ShowWhenClosed | A Boolean value. If TRUE, the threshold is displayed when it is closed. |

Table 9-2 Threshold Outputs

| Output Name | Description |
|----------------|----------------------------------------------------------------------------|
| Open | If open, then return TRUE. Otherwise, return FALSE. |
| OpenFraction | The fraction of total simulation time that the threshold is open. |
| ClosedFraction | The fraction of total simulation time that the threshold is closed. |
| OpenCount | The number of times the threshold's state has changed from closed to open. |
| ClosedCount | The number of times the threshold's state has changed from open to closed. |

The respond of an object to the closure of one of its Thresholds depends on which one of the following keywords was used.

Table 9-3 Keywords that accept Thresholds

| Keyword | Description |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ImmediateThresholdList | A list of thresholds that must be satisfied for the object to operate. Operation is stopped immediately when one of the thresholds closes. If a threshold closes part way through processing an entity, the work is considered to be partly done and the remainder is completed once the threshold re-opens. |
| ImmediateReleaseThresholdList | A list of thresholds that must be satisfied for the object to operate. Operation is stopped immediately when one of the thresholds closes. If a threshold closes part way through processing an entity, the work is interrupted and the entity is released. |
| OperatingThresholdList | A list of thresholds that must be satisfied for the object to operate. If a threshold closes part way through processing an entity, the remaining work is completed and the entity is released before the object is closed. |

A given Threshold can appear in only one of these keywords at a time. A single Threshold can be used by multiple objects.

9.2 Maintenance and Breakdowns

Planned maintenance and breakdowns are modelled using the DowntimeEntity (Section 13.9), which generates random or scheduled events based on either working time or calendar time. Normally, a maintenance activity is scheduled to occur at regular intervals based on calendar time. Breakdowns are normally modelled to occur randomly based on the working time for the object.

The object that will undergo the maintenance or breakdown has a series of keywords that determine how the object responds to a particular maintenance or breakdown event.

Table 9-4 Maintenance and Breakdowns Inputs

| Keyword | Description |
|------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| WorkingStateList | A list of states for which the entity is considered to be working. |
| ImmediateMaintenanceList | A list of DowntimeEntities representing planned maintenance that must be performed immediately, interrupting any work underway at present. |
| ForcedMaintenanceList | A list of DowntimeEntities representing planned maintenance that must begin as soon as task underway at present is finished. |
| OpportunisticMaintenanceList | A list of DowntimeEntities representing planned maintenance that can wait until task underway at present is finished and the queue of tasks is empty. |
| ImmediateBreakdownList | A list of DowntimeEntities representing unplanned maintenance that must be performed immediately, interrupting any work underway at present. |
| ForcedBreakdownList | A list of DowntimeEntities representing unplanned maintenance that must begin as soon as task underway at present is finished. |
| OpportunisticBreakdownList | A list of DowntimeEntities representing unplanned maintenance that can wait until task underway at present is finished and the queue of tasks is empty. |

Only one breakdown or maintenance activity can occur at any given time. If multiple breakdown/maintenance activities are scheduled for the same time, they are performed sequentially.

It is possible under some circumstances for a breakdown or maintenance to occur while a threshold is closed. In this case, the state is set to the appropriate breakdown or maintenance state.

The outputs for maintenance, breakdowns, and thresholds are grouped together as follows.

Table 9-5 Maintenance, Breakdowns, and Thresholds Outputs

| Output Name | Description |
|--------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Open | Returns TRUE if all the thresholds specified by the OperatingThresholdList, ImmediateThresholdList, and ImmediateReleaseThresholdList keywords are open. |
| Working | Returns TRUE if work is being performed. |
| Maintenance | Returns TRUE if maintenance is being performed. |
| Breakdown | Returns TRUE if a breakdown is being repaired. |
| Utilisation | The fraction of calendar time (excluding the initialisation period) that this object is in the Working state. |
| Commitment | The fraction of calendar time (excluding the initialisation period) that this object is in any state other than Idle. |
| Availability | The fraction of calendar time (excluding the initialisation period) that this object is in any state other than Maintenance or Breakdown. |
| Reliability | The ratio of Working time to the sum of Working time and Breakdown time. All times exclude the initialisation period. |

9.3 States

Objects implementing states have the following keywords and outputs.

Table 9-6 States Inputs

| Keyword | Description |
|---------------|---------------------------------------------------------------------------------------------------------------------------------|
| StateGraphics | A list of state/DisplayEntity pairs. For each state, the graphics will be changed to those for the corresponding DisplayEntity. |

Table 9-7 States Outputs

| Output Name | Description |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| State | <p>The present state for this object, used for statistics collection. Typical states are:</p> <ul style="list-style-type: none"> • Working. The object is performing its normal process. • Idle. The object is available for work, but has work to perform. • Stopped. One or more of the Thresholds are closed and there is no maintenance being performed or breakdown being repaired. • Maintenance. One of the DowntimeEntities entered to the Maintenance keywords is active. • Breakdown. One of the DowntimeEntities entered to the Breakdown keywords is active. <p>Additional states can be defined for some objects.</p> |
| WorkingState | Set to true if the present state is one of the pre-defined working states. |
| WorkingTime | Total time spent in any one of the working states. |
| StateTimes | Total time spent in each of the states. |

10 Graphics

10.1 DisplayEntity

The DisplayEntity is the basic 3D graphical object in JaamSim. All JaamSim objects that have graphics are subclasses of DisplayEntity.

All objects intended for visualization in a model display window in JaamSim have a set of basic graphics keywords used to define their appearance. These are found in the Graphics tab of the Input Editor when the object is selected. For polyline type objects the inputs for Position, Alignment, Size, and Orientation are replaced by those for Points and CurveType.

Table 10-1 DisplayEntity Inputs

| Keyword | Description |
|-------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Normal Objects</u> | |
| Position | The location of the object in {x, y, z} coordinates. |
| Alignment | The point within the object that is located at the coordinates of its Position input. Expressed with respect to a unit box centered about { 0 0 0 }. |
| Size | The size of the object in {x, y, z} coordinates. If only the x- and y-dimensions are given then the z-dimension is assumed to be zero. |
| Orientation | Euler angles defining the rotation of the object. |
| <u>Polyline Type Objects</u> | |
| Points | A list of points in {x, y, z} coordinates that define a polyline. When only two coordinates are given it is assumed that z = 0. |
| CurveType | The type of curve interpolation used for line type entities. |
| <u>All Objects</u> | |
| Region | If a Region is specified, the Position and Orientation inputs for the present object will be relative to the Position and Orientation of the specified Region. If the specified Region is moved or rotated, the present object will move to maintain its relative position and orientation. |
| RelativeEntity | If an object is specified, the Position input for the present object will be relative to the Position for the specified object. If the specified object is moved, the present object will move to maintain its relative position. |
| DisplayModel | The graphic representation of the object. If a list of DisplayModels is entered, each one will be displayed provided that its DrawRange input is satisfied. This feature allows the object's appearance to change with its distance from the View window's camera. |
| Show | If TRUE, the object is shown in the View windows. |
| Movable | If TRUE, the object will respond to mouse clicks and can be positioned by dragging with the mouse. |

Table 10-2 DisplayEntity Outputs

| Output Name | Description |
|---------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name | The unique input name for this entity. |
| ObjectType | The class of objects that this entity belongs to. |
| SimTime | The present simulation time. |
| <i>User-defined attributes and custom outputs</i> | The present values for each of the attributes and custom outputs that were defined for the DisplayEntity. (see Section 6.2) |
| Position | The present {x, y, z} coordinates of the DisplayEntity in its region. |
| Size | The present {x, y, z} components of the DisplayEntity's size. |
| Orientation | The present {x, y, z} Euler angles of the DisplayEntity's rotation. |
| Alignment | The present {x, y, z} coordinates of a point on the DisplayEntity that aligns direction with the position output. Each component should be in the range [-0.5, 0.5]. |

10.2 DisplayModel

The graphical appearance of a DisplayEntity and its subclasses is determined by its DisplayModel. The two objects work together to generate an object's display. In general, the DisplayEntity determines what is displayed, while its DisplayModel determines how it is displayed. A number of different subclasses of DisplayModel are available to match the various subclasses of DisplayEntity.

For example, the Text and TextModel objects work together to display text: the text to be displayed is determined by the Text object, while the style of the displayed text (font, bold, italics, colour, etc.) is determined by the TextModel. In this case, the TextModel plays the same role as a text style in word processing software. Users can ensure that the same text style is used for multiple Text objects by sharing the same TextModel between all these objects.

One DisplayModel can be shared between multiple DisplayEntities. This is an essential feature for the case of complex 3D content built from millions of triangles. In this case, a ColladaModel (a subclass of DisplayModel) stores 3D information that can be shared between multiple DisplayEntities. The 3D content is loaded and stored only once, even though it is displayed many times in various locations. Even in the case of animated 3D content, only one ColladaModel is needed to display a different animation state in each location.

Although DisplayModels determine the appearance of a DisplayEntity, the DisplayModel has no graphics itself and therefore cannot be dragged and dropped in the normal manner. To create a new DisplayModel, simply duplicate an existing DisplayModel. JaamSim starts with a pre-defined example of each type of DisplayModel that can be used for this purpose. Duplicate an existing DisplayModel by right-clicking its entry in the Object Selector and selecting Duplicate, then edit its keyword values and name as necessary.

A selection of DisplayModels can be found in the Display Models palette in the Object Selector. Each type of DisplayModel is intended for specific types of DisplayEntities.

Table 10-3 Types of DisplayModel

| DisplayModel | Description | Usage |
|---------------|--------------------------------------------------------------------------------|-------------------------------------------------------------------------------|
| ColladaModel | An imported 3D object. | DisplayEntity and all its sub-classes. |
| ImageModel | An imported picture. | DisplayEntity and all its sub-classes. |
| ShapeModel | A flat geometric object such as a circle or rectangle. | DisplayEntity and all its sub-classes. |
| PolylineModel | A series of line segments connected by nodes. | Selected sub-classes of DisplayEntity such as EntityConveyor and EntityDelay. |
| TextModel | A text style (font, colour, etc.). | Text, OverlayText, BillboardText, and OverlayClock objects. |
| ArrowModel | Similar to PolylineModel except that the last line terminates in an arrowhead. | Arrow object. |
| GraphModel | Formatting inputs for graphs. | Graph object. |

All of these DisplayModel objects have the same Graphics keywords that control optional rendering and scaling at different drawing ranges.

Table 10-4 DisplayModel Inputs

| Keyword | Description |
|--------------|---------------------------------------------------------------------------------------------------------------|
| VisibleViews | A list of Views for which this DisplayModel is shown. If empty, the model appears on all Views. |
| DrawRange | A list of two values for the minimum and maximum distance from the camera this object is visible. |
| ModelScale | A list of three multiplicative factors by which to scale the model in the x, y and z dimensions respectively. |

10.2.1 ColladaModel

ColladaModel objects are used to display custom 3D graphics in a simulation model. The COLLADA file format (.DAE) is an interchange file format used for 3D graphics. Any 3D object such as a DisplayEntity and most of its sub-classes can accept a ColladaModel as its DisplayModel.

A number of other 3D formats can be used in addition to Collada. At the present time, JaamSim supports DAE, OBJ, and JSB formats as well as zipped versions of these files (ZIP). The JSB format is specific to JaamSim:

The JSB format is a binary format that allows complex 3D objects to be loaded much faster than is possible with the DAE and OBJ formats. A JSB file can be exported by right clicking on a ColladaModel in the Object Selector and selecting "Export 3D Binary File (*.jsb)". Note that the exported JSB will look for the same textures as the DAE file and that these should be located in the same relative position to the JSB file as they were to the DAE file.

Managing 3D assets can be complicated because of the large file sizes and the need to provide separate files for the textures. It is recommended that the user place each 3D asset and its texture files in its own ZIP file. This approach greatly reduces the number and size of the files being handled, and ensures that the files can be moved between computers without breaking the file paths.

Note that it is often necessary to edit the DAE file with a text editor, such as Notepad++, to convert any absolute file paths for texture files to relative ones.

Table 10-5 ColladaModel Inputs

| Keyword | Description |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ColladaFile | A file path to the DAE, OBJ, or JSB file to be used for this display model. A ZIP file containing one of these files and its related texture files can also be used, and is the recommended option for this input. The file path must be enclosed in single quotes if it contains spaces. |
| Actions | A list of animation Actions and the object outputs to which they are to be connected, in the format { Action1 Output1 } { Action2 Output2 }. In this example, the actions Action1 and Action2 must be available for the graphical asset specified by the ColladaFile input. The outputs Output1 and Output2 must be valid outputs for the entity to which this ColladaModel is assigned. |

10.2.2 ImageModel

ImageModel objects are used to display custom 2D graphics in a simulation model, such as a picture or a map. Both DisplayEntity and OverlayImage can accept an ImageModel object as an input. Image file types currently supported by JaamSim include BMP, GIF, JPG, PCX and PNG files, or a ZIP file containing any one of these file types.

Table 10-6 ImageModel Inputs

| Keyword | Description |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------|
| ImageFile | A file path to the image file to be used for this display model. Must be enclosed in single quotes if the path contains spaces. |
| Transparent | If TRUE, transparency is enabled for supported image types (GIF and PNG). |
| CompressedTexture | If TRUE, image compression is applied to alleviate memory issues with large images. |

10.2.3 TextModel

TextModel objects specify the general appearance of Text objects and of overlay objects that display text (OverlayText, OverlayClock, and BillboardText). A TextModel object can therefore be used as a style class, with all instances of these Text objects that have the same style sharing the same TextModel.

Table 10-7 TextModel Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------|-----------------------------------------------------|
| FontName, FontColour, FontStyle, DropShadow, DropShadowColour, DropShadowOffset | Same as the keywords used by Text (see Table 11-6). |

10.2.4 Polyline Model

The PolylineModel is used to determine the graphics for entities that appear as a single-segment or multi-segment line, such as EntityConveyor and EntityDelay. It cannot be used by other objects such as a DisplayEntity intended for 3D or 2D graphics.

PolylineModel has no inputs other than the standard ones for DisplayModels.

10.2.5 ArrowModel

The ArrowModel is used to determine the graphics for the Arrow object. It has no inputs other than the standard ones for DisplayModels.

10.2.6 GraphModel

The GraphModel is used to determine the presentation style and proportions for various types of Graph objects. To make the Graph scalable to any size, all length measurements are specified as a fraction of the Graph's total height.

Table 10-8 GraphModel Inputs

| Keyword | Description |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| TitleTextHeight | Text height for the Graph title. |
| XAxisTitleTextHeight | Text height for the x-axis title. |
| YAxisTitleTextHeight | Text height for the y-axis title. |
| LabelTextHeight | The text height for both x-axis and y-axis labels. |
| TitleGap | The gap between the Graph title and the top of the Graph. |
| XAxisTitleGap | The gap between the x-axis labels and the x-axis title. |
| XAxisLabelGap | The gap between the x-axis and its labels. |
| YAxisTitleGap | The gap between the y-axis labels and the y-axis title. |
| YAxisLabelGap | The gap between the y-axis and its labels. |
| TopMargin | Size of the gaps between the respective edges of the outer pane and the Graph. Expressed as a fraction of the Graph's total height. |
| BottomMargin | |
| LeftMargin | |
| RightMargin | |
| TitleTextModel | The TextModel used to determine the font, colour, and style for the Graph title. The dropshadow settings are ignored. Black Verdana text is used if this input is left blank. |
| AxisTitleTextModel | The TextModel used to determine the font, colour, and style for the x-axis and y-axis titles. The dropshadow settings are ignored. Black Verdana text is used if this input is left blank. |
| LabelTextModel | The TextModel used to determine the font, colour, and style for the labels next to the x-axis and y-axis tick marks. The dropshadow settings are ignored. Black Verdana text is used if this input is left blank. |
| GraphColor | The colour of the Graph background. |

| | |
|-----------------|------------------------------------------|
| BackgroundColor | The colour of the outer pane background. |
| BorderColor | The colour of the Graph border. |

10.2.7 ShapeModel

ShapeModel objects are used to display a 2D object whose geometry and colour can be adjusted.

Table 10-9 ShapeModel Inputs

| Keyword | Description |
|---------------|-----------------------------------------------------------------------------------------------------------------|
| Shape | The graphical appearance of the object, chosen from a selection of predefined shapes. |
| FillColor | The colour of the filled part of the DisplayModel, defined by a colour name or RGB values. |
| OutlineColour | The colour of the outline of the DisplayModel, defined by a colour name or RGB values. |
| Filled | If TRUE, the DisplayModel will appear with a solid colour fill. Otherwise, the DisplayModel will appear hollow. |
| Bold | If TRUE, the DisplayModel outline will appear thicker than normal. |

10.3 Importing a 3D Object or Image

The easiest way to create a new DisplayEntity for a 3D object or an image is to use the Import function provided in the Control Panel under File. This method creates a new DisplayEntity that is connected to a new DisplayModel with the imported 3D object or image.

10.4 View Window

The View object is used to hold the position and direction of the camera that creates the 3D image in a View window. A new View can be created by clicking on the View item in the Menu bar. An existing View can be selected for editing by click anywhere in its windows that does not land on an entity.

Table 10-10 View Inputs

| Keyword | Description |
|----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| <u>Graphics</u> | |
| Region | The Region this View is within. |
| ViewCenter | The position the view camera is looking at. |
| ViewPosition | The position the view camera is looking from. |
| WindowSize | The size of the window in pixels (width, height). |
| WindowPosition | The position of the upper left corner of the window in pixels measured from the top left corner of the screen. |
| TitleBarText | Text to place in the title bar of the View window. The window must be closed and re-opened manually after changing the title. |
| ShowWindow | If TRUE, the View window is displayed on-screen. |
| Movable | A Boolean indicating whether the view can be panned or rotated. |
| FollowEntity | The (optional) entity for this view to follow. Setting this input makes the view ignore ViewCenter and interprets ViewPosition as a relative offset to this entity. |
| ScriptedViewPosition | The (optional) scripted curve for the view position to follow. |
| ScriptedViewCenter | The (optional) scripted curve for the view center to follow. |
| SkyboxImage | The image file to use as the background for this view. |

11 Graphics Objects Palette

Graphics Objects are used to create 3D objects, pictures, text, graphs, arrows, and other graphical components needed to visualise and monitor a simulation. The following objects are included in the Graphics Objects palette.

Table 11-1 Graphics Objects Palette

| Object | Description |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Region | Local coordinate system. |
| DisplayEntity | Graphical object displaying either a 3D shape or a 2D picture. |
| Text | Text that appears in the 3D model universe. |
| EntityLabel | Text that labels another object. An EntityLabel can only be created by selecting the "Show Label" option in an object's context menu. Since they cannot be dragged and dropped, EntityLabel does not appear in the Model Builder. |
| InputBox | Text that provides an input value for the model. |
| BarGauge | Gauge that displays the value of an expression. |
| OverlayImage | Picture that appears in a fixed position in the display window. |
| OverlayText | Text that appears in a fixed position in the display window. |
| OverlayClock | Time and date display that appears in a fixed position in the display window. |
| BillboardText | Text that follows a 3D position but is always upright on the screen. |
| Arrow | Line that terminates in an arrow head. |
| Graph | Chart that shows model outputs as they change in time. |
| MimicEntity | Graphical object that copies the appearance of another DisplayEntity. |
| View | Display window showing view of the 3D model universe. A View can only be created using the Views menu in the Menu Bar. |
| VideoRecorder | Object that makes a video recording of the model. |

11.1 Region



The Region object is used to define a local coordinate system. When a Region is specified for an object, its inputs for position and orientation are relative to the position and orientation of its Region. The global coordinate system is the default for objects that do not reference a specific Region.

The Position and Orientation keywords are used to define the origin for the coordinate system and the angles for its coordinate axes. Once these inputs have been set, it is advised to set the inputs for Show and Movable to FALSE so the Region object is no longer visible and cannot be moved accidentally.

Table 11-2 Region Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 11-3 Region Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

11.2 DisplayEntity



The DisplayEntity object is used to add an image or static 3D content in a model. Normally, a DisplayEntity is created automatically when a 3D object or an image is imported through the File > Import menu item.

A DisplayEntity can also be created by drag and drop from the Model Builder. The default appearance of a DisplayEntity is a grey cube. Its appearance can be changed by right-clicking the DisplayEntity and selecting "Change Graphics". The user can then select between the available DisplayModels or can create a new DisplayModel by importing a file in one of the supported 3D graphics formats.

Table 11-4 DisplayEntity Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 11-5 DisplayEntity Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

11.3 Text

abc

The Text object is used to show static or dynamic text in a model. Text objects are used for labelling various parts of the model and for monitoring the status of the model.

The output displayed by the Text object is determined primarily by its Format keyword. Dynamic text can be introduced by including a Java format code such as %s in the Format keyword and specifying the value to be displayed with the DataSource keyword. Some typical Java format codes are %s, which can display both text and numbers, and %.6f, which displays a numeric value with six decimal places.

If the variable text is a number with units, the value can be converted from SI to a specified unit through the Unit keyword.

The easiest way to modify the contents of a Text object is to edit it directly in the view window. Double-click on the object to place it in edit mode. In this mode, text can be entered, deleted, inserted, and highlighted using the same conventions as a typical text editor. Text can be copied from and pasted to the clipboard using Ctrl-C and Ctrl-V. Press the Return key or click on another object to save the changes and return the Text object to its normal mode.

The appearance and style of the text is determined by the keywords under the Font tab, including FontName, FontColour, etc. If no inputs are provided to these keywords, the appearance of the text is determined by the TextModel entered for the DisplayModel keyword. The default TextModel is set to the black Verdana font.

TextModels can be used in the same way as the Text Styles in Microsoft Word. A new TextModel can be created through the following steps:

1. In the Object Selector, right-click on the TextModelDefault object and select Duplicate.
2. Rename the new TextModel object by clicking on its name in the Object Selector, entering the new name.
3. Use the Input Editor to specify the characteristics for the new TextModel, such as font, font style, colour, etc.
4. Use the Input Editor to select the new TextModel for the Text object's DisplayModel keyword.

Table 11-6 Text Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| TextHeight | The height of the font as displayed in the view window. |
| Format | The fixed and variable text to be displayed. If spaces are included, enclose the text in single quotes. If variable text is to be displayed using the DataSource keyword, include the appropriate Java format in the text. For example, %s will display a text output and %.6f will display a number with six decimals of accuracy. |
| UnitType | The unit type for the numerical value to be displayed as variable text. Set to DimensionlessUnit if the variable text is non-numeric such as the state of a Server. |
| Unit | The unit in which to express an expression that returns a numerical value. For example, if the UnitType input has been set to DistanceUnit, then the output value could be displayed in kilometres, instead of meters, by entering km to this keyword. |
| DataSource | An expression that returns the variable text to be displayed. The expression can return a number that will be formatted as text or it can return text directly, such as the state of a Server. |
| FailText | The text to display if there is any failure while formatting the variable text or while evaluating the expression. |
| <u>Font</u> | |
| FontName | The font to be used for the text. |
| FontColour | The colour of the text, specified by a colour keyword or RGB values. |
| FontStyle | The font styles to be applied to the text, e.g. Bold, Italic. |
| DropShadow | If TRUE, then a drop shadow appears for the text. |
| DropShadowColour | The colour for the drop shadow, specified by a colour keyword or RGB values. |
| DropShadOffset | The { x, y, z } coordinates of the drop shadow's offset, expressed as a decimal fraction of the text height. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 11-7 Text Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

11.4 EntityLabel

The EntityLabel object is a special version of a Text object that is used to display the name of a selected object. The only way to create an EntityLabel is to right-click on the object to be labelled and select "Show Label". It cannot be dragged and dropped from the Model Builder.

An EntityLabel moves automatically with the object and is destroyed when the object is destroyed. An object can have only one EntityLabel at a time.

An object name can be changed by editing its EntityLabel in the view window. Double-click on the EntityLabel to enter edit mode and revise the name as desired. Press the Return key or click on another object to accept the new name.

The appearance of all the EntityLabels in a model can be changed by editing the inputs for the EntityLabelModel (under Display Models > TextModel in the Object Selector). The standard colour, font and style (bold and/or italics) for all EntityLabels can be selected in this way.

An EntityLabel has the same inputs and outputs as Text, with the exception of the

Table 11-8 EntityLabel Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| TextHeight | Keywords for Text (see Table 11-6). |
| <u>Font</u> | |
| FontName, FontColour, FontStyle, DropShadow, DropShadowColour, DropShadowOffset | Keywords for Text (see Table 11-6). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 11-9 EntityLabel Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

11.5 InputBox

1.0

The InputBox object is a special version of a Text object that is used to allow a user to enter a model input directly in the view window, without using the Input Editor. The input value can be modified by double-clicking on the displayed value to enter edit mode. After editing is complete, press the Return key or click on another object to accept the new value.

The function of an InputBox object is similar to that of an InputValue object (see Section 13.1). The difference is that InputBox can be used to assign both numerical and non-numerical inputs, while InputValue is restricted to numerical inputs.

Table 11-10 InputBox Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| TextHeight | Keywords for Text (see Table 11-6). |
| TargetInput | The names of the Entity and Keyword that will receive the input. |
| <u>Font</u> | |
| FontName, FontColour, FontStyle, DropShadow, DropShadowColour, DropShadowOffset | Keywords for Text (see Table 11-6). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 11-11 InputBox Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

11.6 BarGauge



The BarGauge object is used to provide a graphical display of the numerical value returned by an expression.

Table 11-12 BarGauge Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| DataSource | Height of the bar expressed as a value between 0 and 1. Values outside this range will be truncated. |
| Colour | Colour of the bar. |
| BackgroundColour | Colour of the gauge's body. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 11-13 BarGauge Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>BarGauge</u> | |
| Value | Value displayed by the gauge. |

11.7 Overlay Objects (OverlayImage, OverlayText, OverlayClock)

Overlay objects are special versions of other objects that are used for graphical display in a simulation model. Unlike other display objects, the position of overlay objects is referenced to the corner of a view window, and so the object does not move when the View is panned or zoomed. These objects are useful for labelling View windows or displaying the model name and company logos. Examples of overlay objects are the default Title and Clock that are provided automatically when a new model is opened.

There are three types of overlay objects, each corresponding to a different graphical object type. The relationship between each overlay object, its parent object type, and its usage is summarized in the following table.

Table 11-14 Overlay Object and Usage Summary

| Overlay Object | Parent Object | Usage |
|----------------|---------------|----------------------------------------------------|
| OverlayImage | DisplayEntity | Static image (Logos, other graphics) |
| OverlayText | Text | Static or dynamic text (Model name, states, rates) |
| OverlayClock | Text | Current time in the simulation model. |

Apart from graphics inputs, each type of overlay object has the same keywords as its parent object. The graphics inputs for overlay objects are shown in the following table.

Table 11-15 Overlay Objects Graphics Inputs

| Keyword | Description |
|----------------|-----------------------------------------------------------------------------------------------------------------------------|
| ScreenPosition | A list of two numbers specifying the spacing, in pixels, between the left and top corner of the View window and the object. |
| AlignRight | If TRUE, the horizontal alignment is referenced to the right side of the View window instead of the left. |
| AlignBottom | If TRUE, the vertical alignment is referenced to the bottom side of the View window instead of the top. |

11.8 BillboardText

A BillboardText object is similar to a Text object, except that the text is always oriented towards the View window and its height is given in pixels instead of metres. BillboardText retains its coordinates in space and in this way differs from the OverlayText object. Both static and dynamic text can be displayed by a BillboardText object using the same keywords as Text objects.

BillboardText is commonly used to label 3D objects. The advantage of using BillboardText is that the label is visible and readable from all view angles.

The keywords for BillboardText are identical to those for Text objects and have the same meaning, except for the TextHeight keyword, which now gives the height of the text in pixels instead of metres. At the present time, this input must include the units of metres (m) even though it is interpreted as pixels. This will be corrected in a future version of the software.

11.9 Arrow



An Arrow object consists of a line, which can be composed of multiple segments, and an arrowhead.

Table 11-16 Arrow Inputs

| Keyword | Description |
|------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| <u>Graphics</u> | |
| Points, CurveType, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |
| Width | The width of the Arrow line segments in pixels. |
| ArrowSize | A set of { x, y, z } numbers that define the size of the arrowhead in those directions at the end of the connector. |
| Colour | The colour of the arrow, defined using a colour keyword or RGB values. |

Table 11-17 Arrow Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

11.10 Graph



The Graph object is a real-time visual representation of one or more quantities, showing their values over a specified time period.

The following figure illustrates the meanings of the keywords that are used to format a Graph.

Figure 11-1 Sample Graph with Formatting Keywords Defined Graphically



Table 11-18 Graph Inputs

| Keyword | Description |
|----------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Key Inputs | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Title | Text for the graph title, enclosed in single quotes if it contains spaces. |
| NumberOfPoints | The number of data points to be displayed on the Graph. This determines the resolution of the graph. |
| UnitType | The type of units for each line specified by the DataSource input. Must be specified before the DataSource input and the Y-Axis inputs. |
| DataSource | One or more sources of data to be graphed against the primary y-axis. Specified as a series of Expressions, each enclosed by braces. |
| LineColours | A list of colours for the data series to be displayed. For multiple colours, each colour must be enclosed in braces as each colour can be itself defined as a list of RGB values. |
| LineWidths | A list of widths, in pixels, for the data series to be displayed. |
| SecondaryUnitType | The type of units for each line specified by the SecondaryDataSource input. Must be specified before the SecondaryDataSource input and the Secondary Y-Axis inputs. |

| | |
|--------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SecondaryDataSource | One or more sources of data to be graphed against the secondary y-axis. Specified as a series of Expressions, each enclosed by braces. |
| SecondaryLineColours | A list of colours for the data series to be displayed. For multiple colours, each colour must be enclosed in braces as each colour can be itself defined as a list of RGB values. |
| SecondaryLineWidths | A list of widths, in pixels, for the data series to be displayed. |
| <u>X-Axis</u> | |
| XAxisTitle | Title text for the x-axis. |
| XAxisUnit | The unit to be used for the x-axis. |
| XAxisStart | The minimum value for the x-axis. |
| XAxisEnd | The maximum value for the x-axis. |
| XAxisInterval | The interval between x-axis labels. |
| XAxisLabelFormat | The Java format to be used for the tick mark values on the x-axis. For example, the format %.1f would display the value 5 as 5.0. |
| XLines | A list of time values between XAxisStart and XAxisEnd where vertical gridlines are inserted. |
| XLinesColor | Colour of the vertical gridlines, or a list corresponding to the colour of each gridline defined in XLines, defined using a colour name or RGB values. |
| <u>Y-Axis</u> | |
| YAxisTitle | Title text for the primary y-axis. |
| YAxisUnit | The unit to be used for the primary y-axis. |
| YAxisStart | The minimum value for the primary y-axis, in units of the DataSource. |
| YAxisEnd | The maximum value for the primary y-axis. |
| YAxisInterval | The interval between primary y-axis labels. |
| YAxisLabelFormat | The Java format to be used for the tick mark values on the primary y-axis. For example, the format %.1f would display the value 5 as 5.0. |
| YLines | A list of values at which to insert horizontal gridlines. |
| YLinesColor | Colour of the horizontal gridlines, defined using a colour name or RGB values. |
| <u>Secondary Y-Axis</u> | |
| SecondaryYAxisTitle | Title text for the secondary y-axis. |
| SecondaryYAxisUnit | The unit to be used for the secondary y-axis. |
| SecondaryYAxisStart | The minimum value for the secondary y-axis. |
| SecondaryYAxisEnd | The maximum value for the secondary y-axis. |
| SecondaryYAxisInterval | The interval between secondary y-axis labels. |
| SecondaryYAxisLabelFormat | The Java format to be used for the tick mark values on the secondary y-axis. |

| | |
|------------------------------------------------------------------------|-----------------------------------------|
| <u>Graphics</u> | |
| Points, CurveType, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 11-19 Graph Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

11.11 MimicEntity



The MimicEntity is used to copy the display of another entity, usually as part of a control panel for a model. The entity whose display is to be copied is specified by the input to the SourceEntity keyword. This input can be the name of a specific entity or an expression that returns an entity.

Table 11-20 MimicEntity Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| SourceEntity | The entity whose graphics are to be copied. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 11-21 MimicEntity Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

11.12 VideoRecorder



The VideoRecorder object is used to create a short video of a model in operation.

The AVI file created by the VideoRecorder is encoded using the VP8 codec, which is NOT supported by Windows Media Player. Furthermore, the present encoding algorithm is quite inefficient making the file size much larger than necessary. Both problems can be solved by re-encoding the video using free open-source software such as HandBrake (www.handbrake.fr).

The original AVI file in VP8 codec can be viewed with the VLC video player (www.videolan.org).

Table 11-22 VideoRecorder Key Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| CustomOutputList | Keyword for User-Defined Variables (see Table 6-14). |
| CaptureStartTime | Simulation time at which to capture the first frame. |
| CaptureInterval | Simulation time between captured frames. |
| CaptureFrames | The total number of frames to capture for the video. The recorded video assumes 30 frames per second. Therefore, if a 2 minute video is required, the number of frames should be set to $120 \times 30 = 3600$. |
| CaptureArea | The size of the video/image, expressed as the number of horizontal and vertical pixels. The top left-hand corner of the captured frames will be the same as the top left-hand corner of the image on the monitor. If the specified image size is larger than the monitor resolution, then the image will be extended beyond the bottom and/or right sides of the monitor. |
| CaptureViews | The list of View windows to be captured. |
| VideoBackgroundColor | The background colour for the captured frames. Only the 3D view portion of the specified windows will be captured. The remainder of the frame, such as the Control Panel or any gaps between the view windows, will be replaced by the background colour. |
| VideoName | A label to append to the run name when the AVI file is saved. The saved file will be named <run name>_<VideoName>.avi. |
| SaveImages | If TRUE, an individual PNG file will be saved for each frame. |
| SaveVideo | If TRUE, an AVI file containing the video will be saved. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 11-23 VideoRecorder Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

12 Probability Distributions Palette

The Probability Distributions palette provides a selection of standard theoretical probability distributions as well as user-defined probability distributions.

Table 12-1 Probability Distributions

| Distribution Name | Description |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| UniformDistribution | Generates samples with a constant probability between minimum and maximum values. |
| TriangularDistribution | Generates samples from a triangular probability distribution between minimum and maximum values. The distribution peaks at its mode. |
| NormalDistribution | Generates samples from a normal probability distribution. |
| ExponentialDistribution | Generates samples from a negative exponential probability distribution. |
| NonStatExponentialDist | Generates samples from a time-varying negative exponential probability distribution. The correct 'non-stationary Poisson process' algorithm is used. |
| ErlangDistribution | Generates samples from an Erlang probability distribution. |
| GammaDistribution | Generates samples from a Gamma probability distribution. |
| BetaDistribution | Generates samples from a Beta probability distribution. |
| WeibullDistribution | Generates samples from a Weibull probability distribution. |
| LogNormalDistribution | Generates samples from a Log-Normal probability distribution. |
| LogLogisticDistribution | Generates samples from a Log-Logistic probability distribution. |
| DiscreteDistribution | Generates samples from a discrete set of values. |
| ContinuousDistribution | Generates samples over a continuous range of values. |
| BooleanSelector | Randomly selects true/false with a user-selectable probability of true. |

Most probability distributions use the following inputs and outputs.

Table 12-2 Distribution Inputs

| Keyword | Description |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UnitType | The unit type for the value returned by the distribution, e.g. TimeUnit. To keep the units consistent for other inputs, this input must be set first. |
| RandomSeed | Seed for the random number generator. Must be an integer greater than or equal to zero. The RandomSeed keyword works together with the GlobalSubstreamSeed under the Simulation object to determine the random sequence. The GlobalSubstreamSeed keyword allows the user to change all the random sequences in a model with a single input. |
| MinValue | The minimum value that can be returned by the distribution. A value less than the minimum is rejected and the distribution is re-sampled. |
| MaxValue | The maximum value that can be returned by the distribution. A value greater than the maximum is rejected and the distribution is re-sampled. |

Table 12-3 Distribution Outputs

| Output Name | Description |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| Value | The last value sampled from the distribution. When used in an expression, this output returns a new sample every time the expression is evaluated. |
| CalculatedMean | The mean value for the distribution calculated directly from the inputs. Ignores the values entered for the MinValue and MaxValue keywords. |
| CalculatedStandardDeviation | The standard deviation for the distribution calculated directly from the inputs. Ignores the values entered for the MinValue and MaxValue keywords. |
| NumberOfSamples | The total number of samples returned by the Probability Distribution. |
| SampleMean | The average of the samples returned by the Probability Distribution. |
| SampleStandardDeviation | The standard deviation of the samples returned by the Probability Distribution. |
| SampleMin | The minimum of the samples returned by the Probability Distribution. |
| SampleMax | The maximum of the samples returned by the Probability Distribution. |

The Probability Distributions were coded using algorithms adapted from "Simulation Modeling & Analysis", 4th Edition, by Averill M. Law. Random numbers for these distributions are generated by the Multiple Recursive Generator developed by L'Ecuyer ("Good Parameters and Implementations for Combined Multiple Recursive Random Number Generators", Operations Res., 47: 159-164 (1999a)).

12.1 Changing the Random Seed

Each Probability Distribution has the RandomSeed keyword, which generates a different pseudo-random sequence of values for each Probability Distribution. Often, a simulation model is executed multiple times, called 'replications', using different random seeds to determine the statistical accuracy of the output parameters.

One way to achieve multiple replications is to change the RandomSeed for each Probability Distribution. However, this can be impractical when large numbers of distributions are used in a model. The GlobalSubstreamSeed keyword for the Simulation object simplifies this process. Changing the input for this keyword causes all Probability Distributions in the model to generate a different random sequence. In effect, the seed for each Probability Distribution is the combination of the inputs to the RandomSeed and GlobalSubstreamSeed keywords.

When multiple objects behave according to a Probability Distribution with the same characteristics, it is advisable to have a unique instance of the Probability Distribution for each object. This avoids any cross-correlation effects arising from the order in which a single distribution may be sampled.

12.2 UniformDistribution



The UniformDistribution object generates random samples with a constant probability between specified minimum and maximum values.

Table 12-4 UniformDistribution Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType, RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-5 UniformDistribution Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.3 TriangularDistribution



The TriangularDistribution object generates random samples from a triangular probability distribution between specified minimum and maximum values. The distribution peaks at its mode.

Table 12-6 TriangularDistribution Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType, RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| Mode | The mode of the triangular distribution, i.e. the value with the highest probability. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-7 TriangularDistribution Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.4 NormalDistribution



The NormalDistribution object generates random samples from a normal probability distribution.

Table 12-8 NormalDistribution Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType, RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| Mean | The mean of the normal distribution (ignoring the MinValue and MaxValue keywords). |
| StandardDeviation | The standard deviation of the normal distribution (ignoring the MinValue and MaxValue keywords). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-9 NormalDistribution Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.5 ExponentialDistribution



The ExponentialDistribution object generates random samples from a negative exponential probability distribution.

Table 12-10 ExponentialDistribution Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType, RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| Mean | The mean of the exponential distribution. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-11 ExponentialDistribution Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.6 NonStatExponentialDist



The NonStatExponentialDist object generates random samples from a time-varying negative exponential probability distribution. The correct "non-stationary Poisson process" algorithm is used.

Table 12-12 NonStatExponentialDist Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| ExpectedArrivals | A time series containing the expected cumulative number of arrivals as a function of time. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-13 NonStatExponentialDist Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.7 ErlangDistribution



The ErlangDistribution object generates random samples from an Erlang probability distribution.

Table 12-14 ErlangDistribution Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType, RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| Mean | The scale parameter for the Erlang distribution. |
| Shape | The shape parameter for the Erlang distribution. An integer value ≥ 1 . Shape = 1 gives the Exponential distribution. For Shape > 10 it is better to use the Gamma distribution. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-15 ErlangDistribution Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.8 GammaDistribution



The GammaDistribution object generates random samples from a Gamma probability distribution.

Table 12-16 GammaDistribution Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType, RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| Mean | The mean of the Gamma distribution. |
| Shape | The shape parameter for the Gamma distribution. A decimal value > 0.0. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-17 GammaDistribution Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.9 BetaDistribution



The BetaDistribution object generates random samples from a Beta probability distribution.

Table 12-18 BetaDistribution Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType, RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| AlphaParam | The alpha tuning parameter. |
| BetaParam | The beta tuning parameter. |
| Scale | The scale parameter for the distribution. This scales the value of the distribution so it return values between 0 and scale. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-19 BetaDistribution Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.10 WeibullDistribution



The WeibullDistribution object generates random samples from a Weibull probability distribution.

Table 12-20 WeibullDistribution Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType, RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| Scale | The scale parameter for the Weibull distribution. |
| Location | The location parameter for the Weibull distribution. |
| Shape | The shape parameter for the Weibull distribution. A decimal value > 0.0. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-21 WeibullDistribution Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.11 LogNormalDistribution



The LogNormalDistribution generates random samples from a Log-Normal probability distribution.

Table 12-22 LogNormalDistribution Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType, RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| Scale | The scale parameter for the Log-Normal distribution. |
| NormalMean | The mean of the dimensionless normal distribution (not the mean of the lognormal). |
| NormalStandardDeviation | The standard deviation of the dimensionless normal distribution (not the standard deviation of the lognormal). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-23 LogNormalDistribution Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.12 LogLogisticDistribution



The LogLogisticDistribution object generates random samples from a Log-Logistic probability distribution.

Table 12-24 LogLogisticDistribution Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType, RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| Scale | The scale parameter for the Log-Logistic distribution. |
| Shape | The shape parameter for the Log-Logistic distribution. A decimal value > 0.0. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-25 LogLogisticDistribution Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.13 DiscreteDistribution



The DiscreteDistribution object generates random samples from a discrete set of values.

Table 12-26 DiscreteDistribution Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType, RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| ValueList | The list of discrete values that can be returned by the distribution. The values can be any positive or negative and can be listed in any order. No interpolation is performed between these values. |
| ProbabilityList | The list of probabilities corresponding to the discrete values in the ValueList. Must sum to 1.0. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-27 DiscreteDistribution Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.14 ContinuousDistribution



The ContinuousDistribution object generates random samples over a continuous range of values.

Table 12-28 ContinuousDistribution Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType, RandomSeed, MinValue, MaxValue | Keywords for Distribution (see Table 12-2). |
| ValueList | The list of values for the user-defined cumulative probability distribution. |
| CumulativeProbabilityList | The list of cumulative probabilities corresponding to the values in the ValueList. The cumulative probabilities must be given in increasing order. The first value must be exactly 0.0. The last value must be exactly 1.0. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-29 ContinuousDistribution Outputs

| Output Name | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Distribution</u> | |
| Value, CalculatedMean, CalculatedStandardDeviation, NumberOfSamples, SampleMean, SampleStandardDeviation, SampleMin, SampleMax | Outputs inherited from Distribution (see Table 12-3). |

12.15 BooleanSelector



The BooleanSelector returns a randomly-selected value of TRUE or FALSE based on a user-specified probability.

Table 12-30 BooleanSelector Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| RandomSeed | Keyword for Distribution (see Table 12-2). |
| TrueProbability | The probability of the Selector returning true. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 12-31 BooleanSelector Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>BooleanSelector</u> | |
| Value | The last value sampled from the distribution. When used in an expression, this output returns a new sample every time the expression is evaluated. |

13 Basic Objects Palette

The Basic Objects palette contains a number of objects that can be used in many types of simulation models. The following objects are provided in the Basic Objects palette.

Table 13-1 Basic Objects Palette

| Object | Description |
|---------------------|-----------------------------------------------------------------------------------------------------|
| InputValue | Provides a way to enter a numerical value directly into the simulation model screen. |
| TimeSeries | Provides a floating point number that changes in simulated time following a series of input values. |
| TimeSeriesThreshold | Specifies a range of values from a TimeSeries for which an activity is permitted. |
| ExpressionThreshold | Specifies a logical condition for which an activity is permitted. |
| BooleanIndicator | Circular entity that changes colour to indicate TRUE and FALSE. |
| ExpressionLogger | Records the values for one or more expressions to a log file at regular intervals. |
| EntitlementSelector | Selects an index on the basis of entitlement from a given set of proportions. |
| ExpressionEntity | Calculates the value for a specified expression. |
| DowntimeEntity | Provides Breakdown and Maintenance controls. |
| ValueSequence | Generates a repeating sequence of numerical values. |
| EventSchedule | Generates a sequence of inter-arrival times from a list of event times. |
| FileToVector | Populates a one-dimensional array with numeric data from a specified file. |
| FileToMatrix | Populates a two-dimensional array with numeric data from a specified file. |
| ScriptEntity | Changes model inputs during a simulation run. |

13.1 InputValue

1.0

The InputValue object is a special version of a Text object that is used to allow a user to enter a numerical input directly in the view window, without using the Input Editor. The input value can be modified by double-click on the displayed value to enter edit mode. After editing is complete, press the Return key or click on another object to accept the new value.

The present value for an InputValue object can be accessed by the inputs to the keywords for other objects either through an expression, e.g. [InputValue1].Value, or by simply entering the name of the InputValue.

The function of an InputValue object is similar to that of an InputBox object (see Section 11.5) except that it is restricted to numerical keywords that can accept an expression or an object that returns a number. In contrast, the InputBox object can be used with any input, both numeric and non-numeric.

Table 13-2 InputValue Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| TextHeight, UnitType | Keywords for Text (see Table 11-6). |
| Value | The numerical value for the input. |
| <u>Font</u> | |
| FontName, FontColour, FontStyle, DropShadow, DropShadowColour, DropShadowOffset | Keywords for Text (see Table 11-6). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 13-3 InputValue Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>InputValue</u> | |
| Value | The present value for this input. |

13.2 TimeSeries



The TimeSeries object simulates a numerical value that varies with time.

Many JaamSim keywords are structured to accept a constant value, a Probability Distribution, a TimeSeries, or an Expression, which makes TimeSeries a powerful and flexible component for building simulation models.

The data for the object consists of a series of time stamps and values specified by the Value keyword. The time stamps can be irregularly spaced and it is possible to repeat the time series over and over again during the simulation using the CycleTime keyword.

The value returned by a TimeSeries object has a specific unit type indicated by the UnitType keyword. To maintain consistency, the UnitType must be specified prior to any other input.

Table 13-4 TimeSeries Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType | The unit type for the time series. The UnitType input must be specified before the Value input. |
| Value | A list of time series records with format { time value }, where: 'time' is the time stamp for the record and 'value' is the time series value. Records are entered in order of increasing simulation time. The appropriate units should be included with both the time and value inputs. The first time stamp MUST be zero simulation time or January 1 00:00:00 of an arbitrary year. If a non-zero year is entered, e.g. '2010-01-01 00:00:00', then the TimeSeries considers this date to be time zero of the simulation and all other timestamps are offset accordingly. |
| CycleTime | The time at which the time series will repeat from the start. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 13-5 TimeSeries Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>TimeSeries</u> | |
| PresentValue | The time series value for the present time. |

Timestamps can be entered in the normal fashion, such as 0 s, 5 h, etc., or in various year/month/day formats (see Table 4-12).

The first timestamp must always be zero seconds (i.e. 0 s) or January 1, 00:00 of an arbitrary year. If a non-zero year is used, for example 2010-01-01, then the TimeSeries considers this date to be time zero of the simulation and all other timestamps are offset accordingly.

The value for the TimeSeries is constant between one timestamp and the next. If the simulation time is greater than the CycleTime, the TimeSeries will repeat from the beginning of the Value list.

For time series with more than a few entries, it is best to put the Value inputs in a separate file and use the Include feature described in Section 8.4. This will require manual editing of the configuration file (see Section 8). In the following example, the include file, TimeSeries1.inc, contains the following entries:

```
TimeSeries1 Value {
{ 0 d 0.00 km/h }
{ 2 d 0.76 km/h }
{ 10 d 0.24 km/h }
}
```

Assuming that the include-file is located in the same directory as the configuration file, the inputs to the configuration file would be:

```
Define TimeSeries { TimeSeries1 }
TimeSeries1 UnitType { SpeedUnit }
Include './TimeSeries1.inc'
TimeSeries1 CycleTime { 14 d }

RecordEdits
```

Note that all the inputs associated with TimeSeries1 appear at the beginning of the configuration file before the RecordEdits statement (see Section 8.6). This position will allow editing of the model in the Input Editor and saving the results without changing the Include file arrangement.

In the example, TimeSeries1 takes on a new value (0.76 km/h) on when simulation time reaches 24 hours and another new value (0.24 km/h) at 240 hours. After 14 days, the TimeSeries completes a cycle and takes on the original value (0 km/h). This 14-day cycle would repeat until the end of the simulation run.

The following inputs in year/month/day format are equivalent to those for the example given above.

```
TimeSeries1 Value {
{ 2014-01-01T00:00:00 0.00 km/h }
{ 2014-01-03T00:00:00 0.76 km/h }
{ 2014-01-11T00:00:00 0.24 km/h }
}
```

13.3 TimeSeriesThreshold



The TimeSeriesThreshold object varies its state between open and closed depending on the present value for a TimeSeries object.

The TimeSeries to be monitored is specified by the TimeSeries keyword. Trigger levels are determined by the MinOpenLimit and MaxOpenLimit keywords. For the TimeSeriesThreshold to be open, the present value for the TimeSeries must be within the range specified by these two keywords.

The LookAhead and Offset keywords provide additional flexibility:

- If a non-zero value is specified for the LookAhead keyword, then the present value for the TimeSeries must be within the range specified by the MinOpenLimit and MaxOpenLimit keywords over the entire LookAhead duration, starting from the present time.
- If a non-zero value is specified for the Offset keyword, then the above rule is modified so that the LookAhead duration begins at the present time plus the offset.

Table 13-6 TimeSeriesThreshold Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType | The unit type for the threshold (e.g. DistanceUnit, TimeUnit, MassUnit). |
| TimeSeries | The TimeSeries object whose values are to be tested. |
| MaxOpenLimit | The largest TimeSeries value for which the threshold is open. The threshold is closed for TimeSeries values greater than MaxOpenLimit. |
| MinOpenLimit | The smallest TimeSeries value for which the threshold is open. The threshold is closed for TimeSeries values smaller than MinOpenLimit. |
| LookAhead | The length of time over which the TimeSeries values must be $\geq \text{MinOpenLimit}$ and $\leq \text{MaxOpenLimit}$. The threshold is open if the TimeSeries values $x(t)$ satisfy $\text{MinOpenLimit} \leq x(t) \leq \text{MaxOpenLimit}$ for simulation times t from $(\text{SimTime} + \text{Offset})$ to $(\text{SimTime} + \text{Offset} + \text{LookAhead})$. |
| Offset | The amount of time that the threshold adds on to every time series lookup. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |
| OpenColour, ClosedColour, ShowWhenOpen, ShowWhenClosed | Keywords for Thresholds (see Table 9-1). |

Table 13-7 TimeSeriesThreshold Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>Threshold</u> | |
| Open, OpenFraction, ClosedFraction, OpenCount, ClosedCount | Outputs inherited from Threshold (see Table 9-2). |

13.4 ExpressionThreshold



The ExpressionThreshold object varies its state between open and closed depending on the value returned by an expression.

The expression to be evaluated is specified by the OpenCondition keyword. It is re-evaluated with every time advance of the model, and on demand when the Open output is tested by another Expression.

Normally, an ExpressionThreshold is open when the OpenCondition is TRUE and IS closed when it is FALSE. However, it is possible to specify a separate expression to close the ExpressionThreshold using the CloseCondition keyword. When this keyword is specified, the threshold remains open until the CloseCondition expression is TRUE.

The CloseCondition keyword introduces some special cases to be addressed:

- If the two conditions conflict by both returning TRUE, then the threshold is considered to be open.
- If both conditions are FALSE, then the threshold's previous open or closed state is retained.
- If both conditions are FALSE at the start of the simulation run, then InitialOpenValue keyword determines whether the threshold is open or closed.

Expression thresholds CANNOT be used in some circumstances. The expressions entered to the OpenCondition and CloseCondition keywords are tested only when an event occurs in the model such as the arrival of an entity or the completion of processing a Server. Normally, this restriction has no effect on a discrete event simulation since every change of the model's state is associated with an event. However, it is possible to enter an expression that changes between TRUE and FALSE without an event occurring. This can occur when simulation time is used explicitly in an expression, e.g. `'this.SimTime > 5[s]'`. An ExpressionThreshold will detect this condition becoming TRUE at the first event that occurs after 5 seconds. The correct way to model this type of condition is to use a TimeSeries and a TimeSeriesThreshold.

The ShowPendingStates keyword is used to detect situations where an ExpressionThreshold is being used inappropriately. A "pending state" is the situation where the present values for the OpenCondition and CloseCondition expressions are inconsistent with the present state of the ExpressionTheshold. This situation can be detected when the ExpressionTheshold is visible in the View window because the OpenCondition and CloseCondition expressions are re-evaluated every time the computer screen is refreshed.

Table 13-8 ExpressionThreshold Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| OpenCondition | The logical condition for the ExpressionThreshold to open. |
| CloseCondition | The logical condition for the ExpressionThreshold to close. If not specified, the CloseCondition defaults to the opposite of the OpenCondition. If the OpenCondition and CloseCondition are both TRUE, then the ExpressionThreshold is set to open. |
| InitialOpenValue | The initial state for the ExpressionThreshold: TRUE = Open, FALSE = Closed. This input is only relevant when the CloseCondition input is used and both the OpenCondition and CloseCondition are FALSE at the start of the simulation run. Otherwise, the initial state is determined explicitly by the OpenCondition and CloseCondition. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |
| OpenColour, ClosedColour, ShowWhenOpen, ShowWhenClosed | Keywords for Thresholds (see Table 9-1). |
| PendingOpenColour | The colour of the ExpressionThreshold graphic when the threshold condition is open, but the gate is still closed. |
| PendingClosedColour | The colour of the ExpressionThreshold graphic when the threshold condition is closed, but the gate is still open. |
| ShowPendingStates | A Boolean value. If TRUE, the ExpressionThreshold displays the pending open and pending closed states. |

Table 13-9 ExpressionThreshold Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>Threshold</u> | |
| Open, OpenFraction, ClosedFraction, OpenCount, ClosedCount | Outputs inherited from Threshold (see Table 9-2). |
| <u>ExpressionThreshold</u> | |
| Open | If open, then return TRUE. Otherwise, return FALSE. |

13.5 BooleanIndicator



The BooleanIndicator allows the state of a specified Expression returning TRUE or FALSE to be displayed.

Table 13-10 BooleanIndicator Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| DataSource | An expression returning a boolean value: zero = FALSE, non-zero = TRUE. |
| TrueColour | The colour of the indicator when the DataSource expression is TRUE. |
| FalseColour | The colour of the indicator when the DataSource expression is FALSE. |
| TrueText | The string returned by the Text output when the DataSource expression is TRUE. |
| FalseText | The string returned by the Text output when the DataSource expression is FALSE. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 13-11 BooleanIndicator Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>BooleanIndicator</u> | |
| Text | If the DataSource expression is TRUE, then return TrueText. If it is FALSE, then return FalseText. |

13.6 ExpressionLogger



The ExpressionLogger object provides the ability to record the value for one or more expressions whenever the object is triggered during the simulation run. An ExpressionLogger can be triggered by one or more types of events:

- At regular time intervals,
- Whenever an object changes state, and
- Whenever the value of an expression changes.

Logging at regular intervals is handled by the keywords in the Key Inputs tab. State and value tracing is handled by the keywords in the Tracing tab.

Table 13-12 ExpressionLogger Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Interval | A fixed interval at which entries will be written to the log file. This input is optional if state tracing or value tracing is specified. |
| UnitTypeList | The unit types for the quantities being logged. Use DimensionlessUnit for text entries. |
| DataSource | One or more sources of data to be logged. Each source is specified by an Expression. Also acceptable are: a constant value, a Probability Distribution, TimeSeries, or a Calculation Object. |
| IncludeInitialization | If TRUE, entries are logged during the initialization period. |
| StartTime | The time for the first log entry. |
| EndTime | The latest time at which to make an entry in the log. |
| <u>Tracing</u> | |
| StateTraceList | A list of entities whose states will be traced. An entry in the log file is made every time one of the entities changes state. Each entity's state is written automatically to the log file - it is not necessary to add an expression to the DataSource keyword's input. |
| ValueUnitTypeList | The unit types for the values being traced. |
| ValueTraceList | One or more sources of data whose values will be traced. An entry in the log file is made every time one of the data sources changes value. Each data source's value is written automatically to the log file - it is not necessary to add an expression to the DataSource keyword's input. Each source is specified by an Expression. Also acceptable are: a constant value, a Probability Distribution, TimeSeries, or a Calculation Object. |
| ValuePrecisionList | The number of decimal places to show for each value in valueTraceList. If only one number is given, then that number of decimal places is used for all values. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 13-13 ExpressionLogger Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

13.7 EntitlementSelector



The EntitlementSelector object is similar to the DiscreteDistribution object in that it returns an index in some range, except that instead of returning a random selection based on probabilities it makes its choice using an entitlement algorithm based on proportions. The entitlement algorithm chooses the index that minimizes the difference between the actual number returned for each index and the expected number based on the proportions.

The ProportionList keyword is used to specify the relative proportions for the N choices.

Table 13-14 EntitlementSelector Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| ProportionList | A list of N numbers equal to the relative proportion for each of the N indices. Must sum to 1.0. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 13-15 EntitlementSelector Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>EntitlementSelector</u> | |
| Value | The last index that was selected. |
| NumberOfSamples | The total number of times that an index has been selected. |
| SampleCount | The number of times each of the N indices has been selected. |
| SampleDifference | The difference between the number of times each index has been selected and the expected number calculated from the proportions. |

13.8 ExpressionEntity



The ExpressionEntity is used to evaluate an expression. It is useful when a complicated expression is used in several different places. Instead of entering the expression several times, it is better to use an ExpressionEntity to evaluate the expression and then use its output Value to pass the result to other expressions.

Table 13-16 ExpressionEntity Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|---------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType | The unit type for the value returned by the expression. |
| Expression | The expression to be evaluated. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 13-17 ExpressionEntity Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>ExpressionEntity</u> | |
| Value | The present value for the expression. |

13.9 DowntimeEntity



The DowntimeEntity object is used to generate planned and unplanned maintenance events for various types of objects. The DowntimeEntity generates the downtime events and their durations, but the objects that use one or more DowntimeEntities must provide their own logic for halting operations.

Table 13-18 DowntimeEntity Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| FirstDowntime | The calendar or working time for the first planned or unplanned maintenance event. If an input is not provided, the first maintenance event is determined by the input for the Interval keyword. A number, an object that returns a number, or an expression can be entered. |
| IntervalWorkingEntity | The object whose working time determines the occurrence of the planned or unplanned maintenance events. Calendar time is used if the input is left blank. |
| DurationWorkingEntity | The object whose working time determines the completion of the planned or unplanned maintenance activity. Calendar time is used if the input is left blank. |
| Interval | The calendar or working time between the start of the last planned or unplanned maintenance activity and the start of the next maintenance activity. A number, an expression, or an object that returns a number can be entered. |
| Duration | The calendar or working time required to complete the planned or unplanned maintenance activity. A number, an expression, or an object that returns a number can be entered. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 13-19 DowntimeEntity Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>DownTimeEntity</u> | |
| StartTime | The time that the most recent downtime event started. |
| EndTime | The time that the most recent downtime event finished or will finish. |
| CalculatedDowntimeRatio | The value calculated directly from model inputs for: (avg. downtime duration)/(avg. downtime interval) |
| Availability | The fraction of calendar time (excluding the initialisation period) during which this type of downtime did not occur. |

13.10 ValueSequence



The ValueSequence object is used to generate a specified sequence of numbers. It can be used instead of a Probability Distribution when a model is to be validated against recorded data. It can also be used to specify an interval or duration for a planned maintenance activity.

The next value in the sequence is returned every time the ValueSequence is referenced. The values are recycled after the last value in the list is returned.

Table 13-20 ValueSequence Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType | The unit type for the generated values. |
| ValueList | The sequence of numbers to be generated. Note that the appropriate unit for the numbers must be entered in the last position. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 13-21 ValueSequence Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>ValueSequence</u> | |
| Index | The position of the last value returned in the list. |
| Value | The last value returned by the ValueSequence. When used in an expression, this output returns a new value every time the expression is evaluated. |

13.11 EventSchedule



The EventSchedule object is similar to the ValueSequence object in that a specified sequence of values is returned. The difference is that an EventSchedule returns the inter-arrival times for a specified sequence of event times. When applied to the InterArrivalTime keyword for an EntityGenerator, it generates a specified sequence of entities at the simulation times provided to the EventSchedule.

Table 13-22 EventSchedule Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| TimeList | A sequence of monotonically-increasing simulation times at which to generate events. If entered in date format, an input of '0000-01-01 00:00:00' corresponds to zero simulation time. |
| CycleTime | Defines when the event times will repeat from the start. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 13-23 EventSchedule Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>ValueSequence</u> | |
| Index | The position of the event time in the list for the last inter-arrival time that was returned. |
| Value | The last inter-arrival time returned from the sequence. When used in an expression, this output returns a new value every time the expression is evaluated. |

13.12 FileToVector and FileToMatrix



The FileToVector and FileToMatrix objects read data contained in a file. The data must be delimited by either spaces or tabs (but NOT commas), and can consist of any mixture of numbers (with or without units, strings, entities, arrays, maps, and lambda functions). Each entry can be any valid expression that is evaluated when the file is loaded, normally at the start of each simulation run. The following table gives some examples of valid data entries:

Table 13-24 Data File Entries

| Entry | Rule | Examples |
|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------|
| Number | If the number includes a unit, the unit must be enclosed by square brackets. If there is a space between the number and the unit, the entry must be enclosed by single quotes. | 1.0 1[m] '1 [m]' |
| String | Strings must be enclosed by both single quotes and double quotes unless an input is provided to the DataFormat keyword. | '"abc"' '"Quick red fox"' |
| Object | Object names are enclosed by square brackets unless an input is provided to the DataFormat keyword. | [Server1] |
| Array | An array must be enclosed by single quotes. | '{1,2,3}' '{1, 2, 3}' |

The use of the DataFormat keyword allows the format rules for strings and objects to be simplified by identifying the type of input in each column of the data file. Furthermore, it permits numbers with the units of time to be entered as timestamps in 'YYYY-MM-DD HH:MM:SS.SSS' or YYYY-MM-DDTHH:MM:SS.SSS format.

Table 13-25 Simplified Format using the DataFormat Keyword

| Entry | DataFormat | Rule | Examples |
|---------------------------|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------|
| Number with units of Time | TIMESTAMP | Time stamps can be entered as either: 'YYYY-MM-DD HH:MM:SS.SSS' or YYYY-MM-DDTHH:MM:SS.SSS See Table 4-12 for more details. | '0000-01-02 06:24:00.0' 0000-01-02T06:24:00.0 |
| String | STRING | Strings can be entered without single or double quotes if they do not include any spaces. Strings containing spaces must be enclosed by single quotes. | abc 'Quick red fox' |
| Object | ENTITY | Object names can be entered without square brackets. | Server1 |
| Any Type | EXPRESSION | The entry must be entered as a valid expression. This default entry is used as a placeholder in the DataFormat list. | 1[m] '"abc"' [Server1] |

The data read from the file is made available to the model through an output named Value. The two objects differ in the type of data returned by this output:

- For FileToVector, the Value output returns a single array combining all the records in the file.
- For FileToMatrix, the Value output returns an array of arrays, with one internal array for each record that was read.

The data file is first read when the simulation is started. It is re-read and the Value output updated whenever the object receives an entity.

Table 13-26 FileToVector and FileToMatrix Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| DataFile | A file containing entries that are delimited by spaces and/or tabs. |
| DataFormat | An optional list of data types that describe the entries in each column of the data file. The data type inputs cause the data file entries to be processed as follows: <ul style="list-style-type: none"> - TIMESTAMP: parse the data entry as a time stamp in 'YYYY-MM-DD HH:MM:SS.SSS' or YYYY-MM-DDTHH:MM:SS.SSS format. - ENTITY: add square bracket around the data entry and parse it as an entity name. - STRING: add double quotes around the data entry and parse it as an string. - EXPRESSION: parse the data entry as an expression. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 13-27 FileToVector and FileToMatrix Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>LinkedComponent</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedComponent (see Table 14-3). |
| FileToVector or FileToMatrix | |
| Value | The data contained in the file. |

13.13 ScriptEntity

Originally developed for video capture, a ScriptEntity can be used to change window Views, to create automatic zooming and panning, and to toggle video capture during a run. Furthermore, keywords defined in the script file can be used to modify simulation and object parameters initially defined in the input configuration file.

At present, the ScriptEntity cannot be dragged and dropped into a model. It can only be created by editing the configuration file (see Section 8).

The ScriptEntity object takes only one keyword, which is the path to a script (.scr) file.

Table 13-28 ScriptEntity Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|---------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Script | Path to the file containing the scripting instructions to be loaded |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 13-29 ScriptEntity Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

The script file contains sets of model inputs preceded by the Time keyword under the ScriptEntity object.

Table 13-30 Inputs for .scr file

| Keyword | Description |
|---------|-----------------------------------------------------------------|
| Time | The simulated time at which the subsequent inputs are executed. |

For example, the following inputs can be entered into a .scr file and then referenced by a ScriptEntity object to slow down the model at a given point in the simulation run:

```
ScriptEntity1 Time          { 24.0 h }
Simulation    RealTime      { TRUE  }
Simulation    RealTimeFactor { 1200 }

ScriptEntity1 Time          { 30.0 h }
Simulation    RealTime      { FALSE }
```

14 Process Flow Palette

The Process Flow palette contains all the objects needed to create process flow type models. These models are characterized by a passive entity that is passed from one object to another following a process flow diagram. These types of models are often used to simulate a manufacturing process where the entities represent parts that are moved between processing stations. The following objects are provided in the Process Flow palette.

Table 14-1 Process Flow Palette

| Object | Description |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------|
| SimEntity | The basic entity for use in a process flow type model. |
| EntityGenerator | Creates copies of a prototype entity at specified intervals. |
| EntitySink | Destroys any entity it receives. |
| Server | Processes a received entity over a specified duration. |
| Queue | Stores received entities until they are needed. |
| EntityConveyor | Transports a received entity along a specified path at a fixed speed. |
| EntityDelay | Delays a received entity by a specified duration. |
| Resource | Set of identical resource units that can be seized and released by various processes. |
| Seize | Seizes one or more units of a Resource. |
| Release | Releases one or more units of a Resource. |
| Assign | Assigns new values to attributes. |
| Branch | Directs a received entity to a selected destination. |
| Duplicate | Sends copies of the received entity to a set of destinations. |
| Combine | Matches entities from multiple queues. The entity from the first queue is passed on while the other entities are destroyed. |
| SetGraphics | Changes the appearance of the received entity. |
| EntityGate | Blocks received entities from progressing further until the EntityGate is opened by one or more Thresholds. |
| EntitySignal | Opens or closes a specified SignalThreshold when an entity is received. |
| SignalThreshold | Threshold that is opened and closed directly by an EntitySignal object. |
| Assemble | Combines sub-components to create an assembled part. |
| EntityContainer | An entity that can hold one or more entities. |
| Pack | Inserts entities in a new EntityContainer. |
| Unpack | Removes all the entities from an EntityContainer which is subsequently destroyed. |
| AddTo | Add entities to an existing EntityContainer. |
| RemoveFrom | Removes some or all of the entities from an EntityContainer. |
| EntityLogger | Records the outputs and state data for a generated entity in an output log. |

| Object | Description |
|------------|-------------------------------------------------|
| Statistics | Collects statistics from the received entities. |

Many of the objects in the Process Flow palette use the inputs and outputs shown in the following tables.

Table 14-2 LinkedComponent Inputs

| Keyword | Description |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DefaultEntity | The default value for the output obj. Normally, obj is set to the last entity received by this object. Prior to receiving its first entity, obj is set to the object provided by DefaultEntity. If an input for DefaultEntity is not provided, then obj is set to null until the first entity is received. |
| NextComponent | The next object to which the processed entity is passed. |
| StateAssignment | The state to be assigned to an entity when it is first received by this object. The state name can be chosen freely by the user. The entity's state is unchanged if the value for this keyword is blank. |

Table 14-3 LinkedComponent Outputs

| Output Name | Description |
|------------------|---------------------------------------------------------------------------------------------------|
| obj | The entity that was received most recently. |
| NumberAdded | The number of entities received from upstream after the initialization period. |
| NumberProcessed | The number of entities processed by this component after the initialization period. |
| NumberInProgress | The number of entities that have been received but whose processing has not been completed yet. |
| ProcessingRate | The number of entities processed per unit time by this component after the initialization period. |
| ReleaseTime | The time at which the last entity was released. |

14.1 SimEntity



The SimEntity object is the basic entity that is passed through a process flow type model. The main feature of SimEntity is that its state can be assigned at various stages of the process flow. The time spent in each state can be accessed using the StateTimes output. Process Flow objects that can receive a SimEntity, such as Server and Queue, can assign a state to the received SimEntity using their StateAssignment keyword.

Table 14-4 SimEntity Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateGraphics | Keyword for StateEntity (see Table 9-6). |
| DefaultStateList | A list of states that will always appear in the output report, even if no time is recorded for this state. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-5 SimEntity Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |

14.2 EntityLauncher



The EntityLauncher object creates an entity when it is clicked by the user or when a specified keyboard key is pressed. An entity can only be generated when the simulation is running in Real Time mode. The EntityLauncher must be selected for the keyboard action to be detected.

The PrototypeEntity keyword identifies the entity to be copied. This entity can be any type of object, no matter how complex. Either a specific object or an expression that returns an object can be entered. The generated copies retain both the graphics of the prototype as well as the values of all its inputs.

The ActionKey input is used to specify a keyboard shortcut for creating an entity. By default, the spacebar is used for this purpose.

Table 14-6 EntityLauncher Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| ActionKey | An optional keyboard key that will cause the EntityLauncher to generate an entity. |
| NextComponent | The next object to which a generated entity is passed. |
| PrototypeEntity | The entity to be copied by the EntityLauncher. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-7 EntityLauncher Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

14.3 EntityGenerator



The EntityGenerator object creates a series of entities that are passed to the next object in a process.

The PrototypeEntity keyword identifies the entity to be copied. This entity can be any type of object, no matter how complex. Either a specific object or an expression that returns an object can be entered. Copies retain both the graphics of the prototype as well as the values of all its inputs.

The rate at which entities are generated is determined by the InterArrivalTime and FirstArrivalTime keywords. These inputs have units of time and can be a constant value, an object that returns a number with units of time (e.g. TimeSeries or Probability Distribution), or an expression that returns such a number.

Table 14-8 EntityGenerator Inputs

| Keyword | Description |
|----------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateGraphics | A list of state/DisplayEntity pairs. For each state, the graphics will be changed to those for the corresponding DisplayEntity. |
| NextComponent | The next object to which a generated entity is passed. |
| FirstArrivalTime | The time at which the first entity is to be generated. Can be a constant value, a TimeSeries, a Probability Distribution, or an Expression. |
| InterArrivalTime | The elapsed time between one generated entity and the next. Can be a constant value, a TimeSeries, a Probability Distribution, or an Expression. |
| EntitiesPerArrival | The number of entities to be generated for each arrival time. Can be a constant value, a TimeSeries, a Probability Distribution, or an Expression. |
| PrototypeEntity | The entity to be copied by the EntityGenerator. |
| MaxNumber | The maximum number of entities to be generated. |
| <u>Thresholds</u> | |
| ImmediateThresholdList, ImmediateReleaseThresholdList, OperatingThresholdList | Keywords for Thresholds (see Table 9-3). |
| <u>Maintenance</u> | |
| WorkingStateList, ImmediateMaintenanceList, ForcedMaintenanceList, ImmediateBreakdownList, ForcedBreakdownList | Keywords for Maintenance and Breakdowns (see Table 9-4). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-9 EntityGenerator Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>StateUserEntity</u> | |
| Open, Working, Maintenance, Breakdown, Utilisation, Commitment, Availability, Reliability | Outputs inherited from StateUserEntity (see Table 9-5). |
| <u>LinkedDevice</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |
| <u>LinkedService</u> | |
| MatchValue | Inherited output that is not relevant for EntityGenerator. |
| <u>EntityGenerator</u> | |
| NumberGenerated | The total number of entities generated, including the initialization period. |
| PresentIAT | The total working time required before the next entity is created. |
| ElapsedTime | The working time that has been completed towards the creation of the next entity. |
| FractionCompleted | The portion of the total working time towards the creation of the next entity that has been completed. |

14.4 EntitySink



The EntitySink object destroys incoming entities.

Table 14-10 EntitySink Key Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-11 EntitySink Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>LinkedDevice</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |

14.5 Server



The Server object processes an incoming entity and then passes it to the next object.

Entities that are waiting to be processed are held by a Queue object identified by the keyword `WaitQueue`. All entities received by the Server first pass through this Queue object, even if the Server is Idle.

Entities can also be sent directly to the Queue. Whenever an entity is added to a Queue, all the Servers that specified this Queue as its `WaitQueue` will be notified. The first Server that is available will then remove the entity for processing.

The rate at which entities are processed is determined by the `ServiceTime` keyword. This input has units of time and accepts a constant value, a `TimeSeries`, a `Probability Distribution`, or an `Expression`.

The Server can be stopped under various circumstances using the `OperatingThresholdList` keyword, which specifies a list of threshold objects such as `SignalThreshold`, `TimeSeriesThreshold`, or `ExpressionThreshold`. All of the specified threshold objects must be open for the Server to operate. However, if a threshold closes while the Server is processing an entity, it will complete its work on that entity before ceasing further operation.

Table 14-12 Server Inputs

| Keyword | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateGraphics | Keyword for StateEntity (see Table 9-6). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| ProcessPosition | The position of the entity being processed relative to the processor. |
| WaitQueue | The queue in which the waiting DisplayEntities will be placed. |
| Match | An expression returning a dimensionless integer value that can be used to determine which of the queued entities is eligible for processing. |
| ServiceTime | The time required to process the incoming entity. Can be a constant value, a TimeSeries, a Probability Distribution, or an Expression. |
| <u>Thresholds</u> | |
| ImmediateThresholdList, ImmediateReleaseThresholdList, OperatingThresholdList | Keywords for Thresholds (see Table 9-3). |
| <u>Maintenance</u> | |
| WorkingStateList, ImmediateMaintenanceList, ForcedMaintenanceList, OpportunisticMaintenanceList, ImmediateBreakdownList, ForcedBreakdownList, OpportunisticBreakdownList | Keywords for Maintenance and Breakdowns (see Table 9-4). |

| | |
|---------------------------------------------------------------------------------------------|-----------------------------------------|
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-13 Server Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>StateUserEntity</u> | |
| Open, Working, Maintenance, Breakdown, Utilisation, Commitment, Availability, Reliability | Outputs inherited from StateUserEntity (see Table 9-5). |
| <u>LinkedDevice</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |
| <u>LinkedService</u> | |
| MatchValue | The present value to be matched to an entity in the queue. |
| <u>Server</u> | |
| ServiceDuration | The total working time required for the present service activity. |
| ServicePerformed | The working time that has been completed for the present service activity. |
| FractionCompleted | The portion of the total service time for the present service activity that has been completed. |

14.6 Queue



A Queue object defines a location for simulation entities to wait for processing by other entities.

Unlike many other objects in this palette, an entity received by a Queue is not passed automatically to the next object. It must wait in the Queue until it is removed by some other object. Queues are used in this way by the Server, Seize, EntityGate, Assemble, Combine, Pack, Unpack, AddTo, and RemoveFrom objects. Whenever an entity is added to a Queue, all the objects that use this Queue are notified that a new entity is available. The first available object will remove the entity from the Queue and start processing it.

Queued entities can be sequenced by an optional priority value specified by the Priority keyword. In most cases, the Priority is specified by an Expression that is evaluated when the entity first arrives at the Queue. Priority is integer valued and decimal values will be truncated, which means, for example, that priority values of 3.2 and 3.6 are identical (i.e. truncated to 3).

Entities with the same priority values can be sequenced in either the default first-in-first-out (FIFO) order, or in last-in-first-out order (LIFO).

Lastly, a queued entity can be provided with an optional identification number using the Match keyword. Objects that use Queues, such as a Server, can request its Queue to provide the first entity with a specified value for the Match keyword. As with the Priority keyword, in most cases the Match value is specified by an Expression that is evaluated when the entity first arrives at the Queue. The Match variable is integer valued, and if a decimal value is provided, it will be truncated.

Table 14-14 Queue Inputs

| Keyword | Description |
|----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateAssignment | Keyword for LinkedComponent (see Table 14-2). |
| Priority | The priority for positioning a received entity in the Queue. Priority is integer valued and a lower numerical value indicates a higher priority. Priority is normally specified by an Expression; however, a dimensionless number or an object that returns a dimensionless number such as a TimeSeries or a Probability Distribution can also be used. |
| Match | An integer value that can be used to label the queued entities. Match is normally specified by an Expression; however, a dimensionless number or an object that returns a dimensionless number such as a TimeSeries or a Probability Distribution can also be used. |
| FIFO | The order in which entities are placed in the queue. TRUE indicates first-in-first-out order (FIFO). FALSE indicates last-in-first-out order (LIFO). |
| RenegadeTime | The time an entity will wait in the queue before deciding whether or not to renege. Evaluated when the entity first enters the queue. A constant value, a distribution to be sampled, a time series, or an expression can be entered. |

| | |
|---------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RenegeCondition | A logical condition that determines whether an entity will renege after waiting for its RenegeTime value. Note that TRUE and FALSE are entered as 1 and 0, respectively. A constant value, a distribution to be sampled, a time series, or an expression can be entered. |
| RenegeDestination | The object to which an entity will be sent if it reneges. |
| Spacing | The amount of graphical space between objects in the Queue. |
| MaxPerLine | Maximum number of objects in each row of the Queue. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-15 Queue Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>LinkedComponent</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedComponent (see Table 14-3). |
| <u>Queue</u> | |
| QueueLength | The present number of objects in the Queue. |
| QueueList | The entities in the queue. |
| QueueTimes | The waiting time for each entity in the queue. |
| PriorityValues | The Priority expression value for each entity in the queue. |
| MatchValues | The Match expression value for each entity in the queue. |
| QueueLengthAverage | The average number of objects in the Queue. |
| QueueLengthStandardDeviation | The standard deviation of the number of objects in the Queue. |
| QueueLengthMinimum | The fewest number of objects in the Queue. |
| QueueLengthMaximum | The largest number of objects in the Queue. |
| QueueLengthTimes | The total time that the queue has length 0, 1, 2, etc. |
| AverageQueueTime | The average time each entity waited in the Queue. Calculated as total accumulated queue time divided by the number of entities added to the Queue. |
| MatchValueCount | The present number of unique match values in the queue. |
| NumberReneged | The number of entities that reneged from the queue. |
| QueuePosition | The position in the queue for an entity undergoing the RenegeCondition test. First in queue = 1, second in queue = 2, etc. |

14.7 EntityConveyor

The EntityConveyor object moves an incoming entity along a path at a fixed speed, and then passes it to the next object.

The travel time for the EntityConveyor is determined by the TravelTime keyword. If a variable travel time is specified through an expression, the conveyor's speed is updated whenever an entity is added to the conveyor or an entity reaches the end of the conveyor.

Table 14-16 EntityConveyor Inputs

| Keyword | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateGraphics | Keyword for StateEntity (see Table 9-6). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| TravelTime | The time required to convey an entity from the start to the end. |
| Width | The width of the line representing the EntityConveyor in pixels. |
| Color | The colour of the line representing the EntityConveyor. |
| <u>Thresholds</u> | |
| ImmediateThresholdList, ImmediateReleaseThresholdList, OperatingThresholdList | Keywords for Thresholds (see Table 9-3). |
| <u>Maintenance</u> | |
| WorkingStateList, ImmediateMaintenanceList, ForcedMaintenanceList, OpportunisticMaintenanceList, ImmediateBreakdownList, ForcedBreakdownList, OpportunisticBreakdownList | Keywords for Maintenance and Breakdowns (see Table 9-4). |
| <u>Graphics</u> | |
| Points, CurveType, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-17 EntityConveyor Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>StateUserEntity</u> | |
| Open, Working, Maintenance, Breakdown, Utilisation, Commitment, Availability, Reliability | Outputs inherited from StateUserEntity (see Table 9-5). |
| <u>LinkedDevice</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |
| <u>LinkedService</u> | |
| MatchValue | Not used by EntityConveyor. |

14.8 EntityDelay



The EntityDelay object delays an incoming entity by a variable duration before passing it to the next object.

The delay is represented as motion along a line that is similar in appearance to the EntityConveyor object. It differs, however, in that the entities moving along the line can pass one another due to their different delay times.

The duration of each entity's delay is determined by the Duration keyword. This input has units of time and accepts a constant value, a TimeSeries, a Probability Distribution, or an Expression.

Table 14-18 EntityDelay Inputs

| Keyword | Description |
|------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateGraphics | Keyword for StateEntity (see Table 9-6). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| Duration | The time required to process the incoming entity. Can be a constant value, a TimeSeries, a Probability Distribution, or an Expression. |
| Animation | If TRUE, entities are moved along the specified path to represent their progression through the delay activity. |
| Width | The width of the line representing the EntityDelay in pixels. |
| Color | The colour of the line representing the EntityDelay. |
| <u>Graphics</u> | |
| Points, CurveType, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-19 EntityDelay Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>LinkedComponent</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedComponent (see Table 14-3). |

14.9 Resource



The Resource object is used to represent a pool of identical equipment or processor units. Resource units can be seized and released by the Seize and Release objects.

The number of resource units is specified using the Capacity keyword, which can accept a constant value or an expression. If the number of units is allowed to vary, it is possible for the number of units in use to be greater than the present value for Capacity. If this situation occurs, the model continues normally and takes resource units out of service one-by-one as they are released. If the Capacity increases, the Resource attempts to make use of the additional capacity immediately.

Table 14-20 Resource Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Capacity | The number of equivalent resource units that are available. The input can be a constant value, a time series, or an expression. |
| StrictOrder | <p>If TRUE, the next entity to seize the resource will be chosen strictly on the basis of priority and waiting time. If this entity is unable to seize the resource because of other restrictions such as an OperatingThreshold input or the unavailability of other resources it needs to seize at the same time, then other entities with lower priority or shorter waiting time will NOT be allowed to seize the resource.</p> <p>If FALSE, the entities will be tested in the same order of priority and waiting time, but the first entity that is able to seize the resource will be allowed to do so.</p> |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-21 Resource Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|---------------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Resource</u> | |
| Capacity | The total number of resource units that can be used. |
| UnitsInUse | The number of units that are in use. |
| AvailableUnits | The number of resource units that are not in use. |
| UnitsSeized | The number of units that have been seized. |
| UnitsReleased | The number of units that have been released. |
| UnitsInUseAverage | The average number of units that have been in use. |
| UnitsInUseStandardDeviation | The standard deviation of the number of units that have been in use. |
| UnitsInUseMinimum | The minimum number of units that have been in use. |
| UnitsInUseMaximum | The maximum number of units that have been in use. |
| UnitsInUseTimes | The total time that the number of resource units in use was 0, 1, 2, etc. |

14.10 Seize



The Seize object allocates one or more units of a specified set of Resources on receiving an incoming entity.

If any of the Resources have insufficient units available, the received entity is directed to a Queue object identified by the WaitQueue keyword. All entities received by the Seize object first pass through this Queue object, even if sufficient units of the Resources are available.

Entities can be sent directly to the Queue specified by the WaitQueue keyword. Whenever an entity is added to the Queue, all the Seize objects that specified this Queue as their WaitQueue will be notified. The first Seize block that has sufficient resource units available will then remove the entity for processing.

Entities waiting for the same Resources at more than one Seize object are processed in order of the priority assigned to them by the input to their Queue's Priority keyword. If the entities in two or more Queues have the same priority value, then the one that has waited the longest is chosen.

All of the specified Resources for the selected entity must be available before any are seized. Once all of them are available, they are seized simultaneously.

Table 14-22 Seize Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateGraphics | Keyword for StateEntity (see Table 9-6). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| WaitQueue | The queue in which the waiting DisplayEntities will be placed. |
| Match | An expression returning a dimensionless integer value that can be used to determine which of the queued entities is eligible for processing. |
| Resource | A list of Resources to be seized. |
| NumberOfUnits | A list containing the number of units of each Resource to be seized. Each entry can be a constant value, a TimeSeries, a Probability Distribution, or an Expression. A decimal input will be truncated to an integer by the internal logic. |
| <u>Thresholds</u> | |
| ImmediateThresholdList, ImmediateReleaseThresholdList, OperatingThresholdList | Keywords for Thresholds (see Table 9-3). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-23 Seize Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>StateUserEntity</u> | |
| Open, Working, Maintenance, Breakdown, Utilisation, Commitment, Availability, Reliability | Outputs inherited from StateUserEntity (see Table 9-5). |
| <u>LinkedDevice</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |
| <u>LinkedService</u> | |
| MatchValue | The present value to be matched to an entity in the queue. |
| <u>Seize</u> | |
| SeizedUnits | The number of resource units seized by the last entity. |

14.11 Release



The Release object de-allocates one or more units of a specified set of Resources on receiving an incoming entity. All of the Resources are released simultaneously.

On release of the Resources, the entities waiting for these Resources are processed in order of the priority assigned to them by the input to their Queue's Priority keyword. If the entities in two or more Queues have the same priority value, then the one that has waited the longest is chosen.

Table 14-24 Release Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateGraphics | Keyword for StateEntity (see Table 9-6). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| Resource | The Resource(s) to be released. |
| NumberOfUnits | The number of units to release from the Resource(s). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-25 Release Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>LinkedComponent</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedComponent (see Table 14-3). |

14.12 Assign



The Assign object performs one or more Attribute assignments whenever it receives an incoming entity. Once the assignments have been performed, the received entity is passed to the next object without delay.

The Assign object is the only place where an Attribute's value can be modified. The Assign object can modify the value of any attribute in the model: its own attributes, the received entity's attributes, or any other object's attributes.

The attribute assignments to be performed are specified by the input to the AttributeAssignmentList keyword. Each assignment has the following form:

```
{ <left-hand side expression> = <right-hand side expression> }
```

The right side of each assignment equation is an expression to be evaluated. The left side is an expression that identifies the attribute whose value is to be modified. For example, if object Assign1 has a dimensionless numerical attribute A, then the following input to its AttributeAssignmentList keyword causes the attribute to be increased by 1:

```
{ 'this.A = this.A + 1' }
```

The Assign object attribute obj can be used to access the attributes of the received entity. For example, if the received entity has an attribute B whose value is a string, then the following input to the AttributeAssignmentList keyword causes this attribute to take the value 'New String':

```
{ 'this.obj.B = [[New String]]' }
```

If another object, say Server1, has an attribute C whose value is a number with units of distance, then the following input to the AttributeAssignmentList keyword causes this attribute to be increased by 1 kilometre:

```
{ '[Server1].C = [Server1].C + 1[km]' }
```

An unlimited number of assignments can be performed by one Assign object.

Table 14-26 Assign Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| AttributeAssignmentList | A list of attribute assignments that are triggered when an entity is received. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-27 Assign Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>LinkedComponent</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedComponent (see Table 14-3). |

14.13 Branch



The Branch object directs an incoming entity to a destination that is chosen from a list of alternatives.

The value for the Choice keyword determines the destination that is chosen: 1 = first branch, 2 = second branch, etc. The input to Choice can be a constant value, a DiscreteDistribution, a TimeSeries, or an Expression. The use of an Expression allows the choice to be made based on the Attributes of the incoming entity.

Table 14-28 Branch Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| DefaultEntity, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| NextComponentList | A list of possible objects to which the processed DisplayEntity can be passed. |
| Choice | A number that determines the choice of next component: 1 = first branch, 2 = second branch, etc. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-29 Branch Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>LinkedComponent</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedComponent (see Table 14-3). |

14.14 Duplicate



The Duplicate object sends copies of the received entity to one or more objects.

Table 14-30 Duplicate Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| TargetComponentList | A list of the objects to which a copy of the received entity will be sent. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-31 Duplicate Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>LinkedComponent</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedComponent (see Table 14-3). |

14.15 Combine



The Combine object takes one entity each from multiple queues and passes on a single entity to the next object. Each of the entities must have the same match value calculated by the input to Match keyword for its Queue. Entities are combined when each queue has at least one entity with the same Match value as the other Queues. When a match is found, the entity in the first Queue is passed to the object specified by the NextComponent keyword, while the entities from the other Queues are destroyed.

Table 14-32 Combine Inputs

| Keyword | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| ProcessPosition | The position of the entity being processed relative to the position of the Combine object. |
| ServiceTime | The time required to process the incoming entity. Can be a constant value, a TimeSeries, a Probability Distribution, or an Expression. |
| WaitQueueList | A list of Queue objects that hold the entities waiting to be combined. |
| RetainAll | If TRUE, all the matching entities are passed to the next component. If FALSE, only the entity in the first queue is passed on. |
| <u>Thresholds</u> | |
| ImmediateThresholdList, ImmediateReleaseThresholdList, OperatingThresholdList | Keywords for Thresholds (see Table 9-3). |
| <u>Maintenance</u> | |
| WorkingStateList, ImmediateMaintenanceList, ForcedMaintenanceList, OpportunisticMaintenanceList, ImmediateBreakdownList, ForcedBreakdownList, OpportunisticBreakdownList | Keywords for Maintenance and Breakdowns (see Table 9-4). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-33 Combine Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>StateUserEntity</u> | |
| Open, Working, Maintenance, Breakdown, Utilisation, Commitment, Availability, Reliability | Outputs inherited from StateUserEntity (see Table 9-5). |
| <u>LinkedDevice</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |
| <u>LinkedService</u> | |
| MatchValue | The present value to be matched to an entity in the queue. |

14.16 SetGraphics



The SetGraphic object is used to change the graphical appearance of a specified entity.

Table 14-34 SetGraphics Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| TargetEntity | The entity whose graphics are to be changed. Defaults to the entity that was received. |
| GraphicsList | List of entities whose graphics can be chosen for assignment to the target entity. |
| Choice | A number that determines the choice of entities from the GraphicsList: 1 = first entity's graphics, 2 = second entity's graphics, etc. A constant value, a distribution to be sampled, or a time series can be entered. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-35 SetGraphics Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>LinkedComponent</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedComponent (see Table 14-3). |

14.17 EntityGate



The EntityGate object allows incoming entities to be blocked temporarily.

When the EntityGate is open and the Queue is empty, incoming entities pass through without delay. When it is closed, incoming entities are directed to a Queue where they are held until the EntityGate becomes Open. When the EntityGate opens, the queued entities are released one at a time, separated by a delay determined by the ReleaseDelay keyword. This delay does not apply to entities that arrive when the EntityGate is open. However, if queued entities are still being released, an incoming entity is placed at the end of the Queue even though the EntityGate is Open.

The EntityGate's state, either Open or Closed, is determined by the input to the OperatingThresholdList keyword, which specifies a list of threshold objects such as SignalThreshold, TimeSeriesThreshold, or ExpressionThreshold. All of the specified threshold objects must be open for the EntityGate to become Open.

Table 14-36 EntityGate Inputs

| Keyword | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| ProcessPosition | The position of the entity being processed relative to the processor. |
| WaitQueue | The queue in which the waiting DisplayEntities will be placed. |
| Match | An expression returning a dimensionless integer value that can be used to determine which of the queued entities is eligible for processing. |
| ReleaseDelay | The time required to remove each entity from the Queue. |
| <u>Thresholds</u> | |
| ImmediateThresholdList, ImmediateReleaseThresholdList, OperatingThresholdList | Keywords for Thresholds (see Table 9-3). |
| <u>Maintenance</u> | |
| WorkingStateList, ImmediateMaintenanceList, ForcedMaintenanceList, OpportunisticMaintenanceList, ImmediateBreakdownList, ForcedBreakdownList, OpportunisticBreakdownList | Keywords for Maintenance and Breakdowns (see Table 9-4). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-37 EntityGate Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>StateUserEntity</u> | |
| Open, Working, Maintenance, Breakdown, Utilisation, Commitment, Availability, Reliability | Outputs inherited from StateUserEntity (see Table 9-5). |
| <u>LinkedDevice</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |
| <u>LinkedService</u> | |
| MatchValue | The present value to be matched to an entity in the queue. |

14.18 EntitySignal



The EntitySignal object sets the state of a SignalThreshold object when it receives an incoming entity.

The SignalThreshold and its new state are specified by the TargetSignalThreshold and NewState keywords, respectively.

Table 14-38 EntitySignal Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| TargetSignalThreshold | The SignalThreshold object whose state will be changed by this EntitySignal. |
| NewState | The state to be set for the SignalThreshold: TRUE if it is open, FALSE if it is closed. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-39 EntitySignal Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>LinkedComponent</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |

14.19 SignalThreshold



The SignalThreshold object varies its state between open and closed when instructed to do so by one or more EntitySignal objects.

SignalThreshold has no internal logic of its own for changing state.

Table 14-40 SignalThreshold Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| InitialState | The initial state for the SignalThreshold at the start of the run. TRUE = open, FALSE = closed. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |
| OpenColour, ClosedColour, ShowWhenOpen, ShowWhenClosed | Keywords for Thresholds (see Table 9-1). |

Table 14-41 SignalThreshold Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>Threshold</u> | |
| Open, OpenFraction, ClosedFraction, OpenCount, ClosedCount | Outputs inherited from Thresholds (see Table 9-2). |

14.20 Assemble



The Assemble object combines a number of sub-components into an assembled part.

Sub-components waiting for processing are collected into a series of Queue objects, identified by the WaitQueueList keyword, one for each type of sub-component.

Incoming sub-components must be sent directly to the appropriate queue, not to the Assemble object itself. If necessary, a Branch object can be used to direct incoming entities to the various queues.

The assembly process begins when there is at least one sub-component entity in each of the Queues. When this occurs, the first sub-component in each Queue is removed and destroyed, and a new assembled part is created by copying the object specified by the PrototypeEntity keyword. The time required to complete the assembly process is given by the ServiceTime keyword.

Table 14-42 Assemble Inputs

| Keyword | Description |
|-------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateGraphics | Keyword for StateEntity (see Table 9-6). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| ProcessPosition | The position of the entity being processed relative to the position of the Assemble object. |
| ServiceTime | The service time required to perform the assembly process. |
| WaitQueueList | A list of Queue objects in which to place the arriving sub-component entities. |
| NumberRequired | The number of entities required from each queue for the assembly process to begin. The last value in the list is used if the number of queues is greater than the number of values. |
| MatchRequired | If TRUE, the all entities used in the assembly process must have the same Match value. The match value for an entity determined by the Match keyword for each queue. The value is calculated when the entity first arrives at its queue. |
| PrototypeEntity | The prototype for entities representing the assembled part. |
| <u>Thresholds</u> | |
| ImmediateThresholdList, ImmediateReleaseThresholdList, OperatingThresholdList | Keywords for Thresholds (see Table 9-3). |

| | |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------|
| <u>Maintenance</u> | |
| WorkingStateList, ImmediateMaintenanceList, ForcedMaintenanceList, OpportunisticMaintenanceList, ImmediateBreakdownList, ForcedBreakdownList, OpportunisticBreakdownList | Keywords for Maintenance and Breakdowns (see Table 9-4). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-43 Assemble Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>StateUserEntity</u> | |
| Open, Working, Maintenance, Breakdown, Utilisation, Commitment, Availability, Reliability | Outputs inherited from StateUserEntity (see Table 9-5). |
| <u>LinkedDevice</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |
| <u>LinkedService</u> | |
| MatchValue | The present value to be matched to an entity in the queue. |

14.21 EntityContainer



The EntityContainer object is used to store a collection of entities for subsequent processing as a unit.

EntityContainer is a sub-class of SimEntity and consequently includes the State output. It can be passed to any object that can accept a SimEntity. The Pack object is used to place entities in generated EntityContainers. The Unpack object is used to remove the entities and to destroy the EntityContainer.

Table 14-44 EntityContainer Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| DefaultStateList | A list of states that will always appear in the output report, even if no time is recorded for this state. |
| PositionOffset | The position of the first entity in the container relative to the container. |
| Spacing | The amount of graphical space shown between entities in the container. |
| MaxPerLine | The number of entities in each row inside the container. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-45 EntityContainer Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>EntityContainer</u> | |
| Count | The present number of entities in the EntityContainer. |
| EntityList | The entities contained by the EntityContainer. |

14.22 Pack



The Pack object is used to generate EntityContainers and fill them with incoming entities.

EntityContainers are created automatically on demand by the Pack object. The number of entities to be packed in each EntityContainer is determined by the NumberOfEntities keyword. Once the specified number of entities is available in the Queue, the entities are packed one by one in the EntityContainer in the same order as the Queue. The time to pack each entity is specified by the ServiceTime keyword.

Table 14-46 Pack Key Inputs

| Keyword | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateGraphics | Keyword for StateEntity (see Table 9-6). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| ProcessPosition | The position of the entity being processed relative to the processor. |
| WaitQueue | The queue in which the waiting DisplayEntities will be placed. |
| Match | An expression returning a dimensionless integer value that can be used to determine which of the queued entities is eligible for processing. |
| PrototypeEntityContainer | The prototype for EntityContainers to be generated. The generated EntityContainers will be copies of this entity. |
| NumberOfEntities | The number of entities to pack into the container. |
| ServiceTime | The service time required to pack each entity in the container. |
| <u>Thresholds</u> | |
| ImmediateThresholdList, ImmediateReleaseThresholdList, OperatingThresholdList | Keywords for Thresholds (see Table 9-3). |
| <u>Maintenance</u> | |
| WorkingStateList, ImmediateMaintenanceList, ForcedMaintenanceList, OpportunisticMaintenanceList, ImmediateBreakdownList, ForcedBreakdownList, OpportunisticBreakdownList | Keywords for Maintenance and Breakdowns (see Table 9-4). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-47 Pack Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>StateUserEntity</u> | |
| Open, Working, Maintenance, Breakdown, Utilisation, Commitment, Availability, Reliability | Outputs inherited from StateUserEntity (see Table 9-5). |
| <u>LinkedDevice</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |
| <u>LinkedService</u> | |
| MatchValue | The present value to be matched to an entity in the queue. |
| <u>Pack</u> | |
| Container | The EntityContainer that is being filled. |

14.23 Unpack



The Unpack object is used to remove the entities from incoming EntityContainers.

Entities are unpacked one by one from the EntityContainer in the same order as they were packed. The time to unpack each entity is specified by the ServiceTime keyword. The received EntityContainer is destroyed once it has been unpacked.

Table 14-48 Unpack Inputs

| Keyword | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateGraphics | Keyword for StateEntity (see Table 9-6). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| ProcessPosition | The position of the entity being processed relative to the processor. |
| WaitQueue | The queue in which the waiting DisplayEntities will be placed. |
| Match | An expression returning a dimensionless integer value that can be used to determine which of the queued entities is eligible for processing. |
| ServiceTime | The service time required to unpack each entity. |
| <u>Thresholds</u> | |
| ImmediateThresholdList, ImmediateReleaseThresholdList, OperatingThresholdList | Keywords for Thresholds (see Table 9-3). |
| <u>Maintenance</u> | |
| WorkingStateList, ImmediateMaintenanceList, ForcedMaintenanceList, OpportunisticMaintenanceList, ImmediateBreakdownList, ForcedBreakdownList, OpportunisticBreakdownList | Keywords for Maintenance and Breakdowns (see Table 9-4). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-49 Unpack Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>StateUserEntity</u> | |
| Open, Working, Maintenance, Breakdown, Utilisation, Commitment, Availability, Reliability | Outputs inherited from StateUserEntity (see Table 9-5). |
| <u>LinkedDevice</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |
| <u>LinkedService</u> | |
| MatchValue | The present value to be matched to an entity in the queue. |

14.24 AddTo



The AddTo object is used to add a given number of entities to a received EntityContainer.

The number of entities to be added to each EntityContainer is determined by the NumberOfEntities keyword. Once the specified number of entities is available in the WaitQueue and there is an EntityContainer waiting in the ContainerQueue, the entities are added one by one to the EntityContainer in the same order as they appeared in the WaitQueue. The time to add each entity is specified by the ServiceTime keyword. EntityContainers are filled in the order in which they appear in the ContainerQueue.

When an input to the Match keyword is provided, only the entities in the WaitQueue that have the same Match value are added to the EntityContainer. The Match keyword for the ContainerQueue is not used by this logic.

Table 14-50 AddTo Inputs

| Keyword | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateGraphics | Keyword for StateEntity (see Table 9-6). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| ProcessPosition | The position of the entity being processed relative to the processor. |
| WaitQueue | The queue in which the waiting DisplayEntities will be placed. |
| Match | An expression returning a dimensionless integer value that can be used to determine which of the queued entities is eligible for processing. |
| NumberOfEntities | The number of entities to pack into the container. |
| ServiceTime | The service time required to pack each entity in the container. |
| ContainerQueue | The Queue in which the arriving EntityContainers are stored while they wait for processing. |
| <u>Thresholds</u> | |
| ImmediateThresholdList, ImmediateReleaseThresholdList, OperatingThresholdList | Keywords for Thresholds (see Table 9-3). |
| <u>Maintenance</u> | |
| WorkingStateList, ImmediateMaintenanceList, ForcedMaintenanceList, OpportunisticMaintenanceList, ImmediateBreakdownList, ForcedBreakdownList, OpportunisticBreakdownList | Keywords for Maintenance and Breakdowns (see Table 9-4). |

| | |
|---------------------------------------------------------------------------------------------|-----------------------------------------|
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-51 AddTo Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>StateUserEntity</u> | |
| Open, Working, Maintenance, Breakdown, Utilisation, Commitment, Availability, Reliability | Outputs inherited from StateUserEntity (see Table 9-5). |
| <u>LinkedDevice</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |
| <u>LinkedService</u> | |
| MatchValue | The present value to be matched to an entity in the queue. |
| <u>Pack</u> | |
| Container | The EntityContainer that is being filled. |

14.25 RemoveFrom



The RemoveFrom object is used to remove a given number of entities from an incoming EntityContainer.

Entities are removed one by one from the EntityContainer in the same order as they were packed. The time to unpack each entity is specified by the ServiceTime keyword. EntityContainers are passed to another object once the specified number of entities have been unpacked.

Table 14-52 RemoveFrom Inputs

| Keyword | Description |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| StateGraphics | Keyword for StateEntity (see Table 9-6). |
| DefaultEntity, NextComponent, StateAssignment | Keywords for LinkedComponent (see Table 14-2). |
| ProcessPosition | The position of the entity being processed relative to the processor. |
| WaitQueue | The queue in which the waiting DisplayEntities will be placed. |
| Match | An expression returning a dimensionless integer value that can be used to determine which of the queued entities is eligible for processing. |
| ServiceTime | The service time required to unpack each entity. |
| NumberOfEntities | The maximum number of entities to remove from the container. |
| NextForContainers | The next object to which the processed EntityContainer is passed. |
| <u>Thresholds</u> | |
| ImmediateThresholdList, ImmediateReleaseThresholdList, OperatingThresholdList | Keywords for Thresholds (see Table 9-3). |
| <u>Maintenance</u> | |
| WorkingStateList, ImmediateMaintenanceList, ForcedMaintenanceList, OpportunisticMaintenanceList, ImmediateBreakdownList, ForcedBreakdownList, OpportunisticBreakdownList | Keywords for Maintenance and Breakdowns (see Table 9-4). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-53 RemoveFrom Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>StateUserEntity</u> | |
| Open, Working, Maintenance, Breakdown, Utilisation, Commitment, Availability, Reliability | Outputs inherited from StateUserEntity (see Table 9-5). |
| <u>LinkedDevice</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedDevice (see Table 14-3). |
| <u>LinkedService</u> | |
| MatchValue | The present value to be matched to an entity in the queue. |

14.26 EntityLogger



The EntityLogger object records the outputs, attributes, and state data for each entity that it receives.

The output log file is created automatically when the simulation run begins. The output file is named <configuration file name>-<EntityLogger name>.log to ensure that it is unique for the simulation run. For example, if the configuration file is named "run1.cfg" and the EntityLogger's name is EntityLogger1, then the name of the log file will be "run1-EntityLogger1.log". A pre-existing file with this name will be overwritten once the simulation run is started.

Table 14-54 EntityLogger Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitTypeList | The unit types for the quantities being logged. Use DimensionlessUnit for text entries. |
| DataSource | One or more sources of data to be logged. Each source is specified by an Expression. Also acceptable are: a constant value, a Probability Distribution, TimeSeries, or a Calculation Object. |
| IncludeInitialization | If TRUE, entries are logged during the initialization period. |
| StartTime | The time for the first log entry. |
| EndTime | The latest time at which to make an entry in the log. |
| NextComponent | Keyword for LinkedComponent (see Table 14-2). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-55 EntityLogger Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>Logger</u> | |
| LogTime | The simulation time at which the last log entry was made. |
| <u>EntityLogger</u> | |
| obj | The entity that was received most recently. |

14.27 Statistics



The Statistics object collects statistical information on the entities it receives.

The quantity to be tracked is specified using the SampleValue keyword, which accepts an Expression. Some example inputs are:

- 'this.obj.A' - the sample value is the Attribute 'A' carried by the received entity
- 'this.SimTime - this.obj.t' - the sample value is the simulation time that has elapsed since the Attribute 't' for the received entity was set to the simulation time at some earlier point in the model

Table 14-56 Statistics Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|--------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| NextComponent | Keyword for LinkedComponent (see Table 14-2). |
| UnitType | The unit type for the variable whose statistics will be collected. |
| SampleValue | The variable for which statistics will be collected. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 14-57 Statistics Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>StateEntity</u> | |
| State, WorkingState, WorkingTime, StateTimes | Outputs inherited from StateEntity (see Table 9-7). |
| <u>LinkedComponent</u> | |
| obj, NumberAdded, NumberProcessed, NumberInProgress, ProcessingRate, ReleaseTime | Outputs inherited from LinkedComponent (see Table 14-3). |
| <u>Statistics</u> | |
| SampleMinimum | The smallest value that was recorded. |
| SampleMaximum | The largest value that was recorded. |
| SampleAverage | The average of the values that were recorded. |
| SampleStandardDeviation | The standard deviation of the values that were recorded. |
| StandardDeviationOfTheMean | The estimated standard deviation of the sample mean. |
| TimeAverage | The average of the values recorded, weighted by the duration of each value. |
| TimeStandardDeviation | The standard deviation of the values recorded, weighted by the duration of each value. |

15 Calculation Objects Palette

The Calculation Objects Palette contains objects for building continuous type models and mixed discrete-event/continuous models. As the name suggests, a continuous model changes state continuously as time advances. JaamSim is able to model this type of behaviour very efficiently using outputs, Attributes, and Expressions. The following objects are provided in the Calculation Objects palette.

Table 15-1 Calculation Objects Palette

| Object | Description |
|----------------|--------------------------------------------------------------------------------|
| Controller | Signals the updating of each component in the specified sequence. |
| WeightedSum | Calculates the weighted sum of the input values. |
| Polynomial | Evaluates a polynomial function of the input value. |
| Integrator | Integrates the input value over time. |
| Differentiator | Differentiates the input value over time. |
| PIDController | Proportional-Integral-Differential controller. |
| Lag | Calculates the LAG operation for the input value. |
| MovingAverage | Calculates a moving average of the input value over a specified range of time. |
| SineWave | Generates a sinusoidal wave. |
| SquareWave | Generates a square wave. |
| UnitDelay | Delays the input value by one Controller time step. |

Many calculation objects use the following inputs and outputs.

Table 15-2 Calculation Object Inputs

| Keyword | Description |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Controller | The Controller object that signals the updating of the calculation. |
| SequenceNumber | The sequence number used by the Controller to determine the order in which calculations are performed. A calculation with a lower value is executed before one with a higher value. |
| UnitType | The unit type for the input value(s) to the calculation. |
| InputValue | The input value for the present calculation. |

Table 15-3 Calculation Object Outputs

| Output Name | Description |
|-------------|----------------------------------------------------|
| Value | The result of the calculation at the present time. |

15.1 Controller



The Controller object generates the update signals for the individual objects in a continuous type model that are managed by this Controller.

Table 15-4 Controller Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| SamplingTime | Interval between update signals to the objects managed by this Controller. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 15-5 Controller Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

15.2 WeightedSum



The WeightedSum object adds two or more input values. A set of dimensionless constants can be provided to multiply each input value prior to addition.

$$y = C_1 * x_1 + C_2 * x_2 + \dots + C_N * x_N, \text{ if the coefficients } C_i \text{ are specified}$$

$$= x_1 + x_2 + \dots + x_N, \text{ if the coefficients } C_i \text{ are not specified}$$

where:

y = present output value for the weighted sum
 x_i = present value for the i^{th} input to the weighted sum
 C_i = i^{th} entry in the **CoefficientList** input
 N = number of inputs to the weighted sum

The value returned by the WeightedSum is calculated on demand.

Table 15-6 WeightedSum Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType | The unit type for the inputs to the weighted sum and for the value returned. |
| InputValueList | The list of inputs to the weighted sum. All inputs must have the same unit type. |
| CoefficientList | The list of dimensionless coefficients to be applied to the input values. If left blank, the input values are simply added without applying any coefficients. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 15-7 WeightedSum Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>WeightedSum</u> | |
| Value | The calculated value for the weighted sum. |

15.3 Polynomial



The Polynomial object evaluates a specified polynomial for the present input value:

$$y = C_0 + C_1*x + C_2*x^2 + \dots + C_N*x^N$$

where:

y = present output value for the polynomial
 x = present input to the polynomial
 C_i = i^{th} entry in the **CoefficientList** input

The value returned by the Polynomial is calculated on demand.

Table 15-8 Polynomial Key Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| InputValue | The dimensionless input value to the polynomial |
| CoefficientList | The list of dimensionless coefficients for the polynomial function. The number of coefficients provided determines the number of terms in the polynomial. For example, inputs c0, c1, c2 specifies the second order polynomial $P(x) = c0 + c1*x + c2*x^2$. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 15-9 Polynomial Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>WeightedSum</u> | |
| Value | The calculated value for the polynomial. |

15.4 Integrator



The Integrator object integrates the input value with respect to time using the trapezoidal rule:

$$y = Y + (t - T) * 0.5 * (x + X)$$

where:

y = present output value for the integrator
 x = present input to the integrator
 t = present simulation time
 Y = output value for the integrator at the last update time
 X = input to the integrator at the last update time
 T = simulation time at the last update

The value returned by the Integrator is calculated on demand. The update signal received from the Controller is used only to record the values Y , X , and T used in the calculation.

Table 15-10 Integrator Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Controller, SequenceNumber, UnitType, InputValue | Keywords for Calculation Objects (see Table 15-2). |
| InitialValue | The initial value for the integral at time = 0. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 15-11 Integrator Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>DoubleCalculation</u> | |
| Value | Output inherited from Calculation Object. |

15.5 Differentiator



The Differentiator object calculates the time derivative of the input value:

$$y = \frac{(x - X)}{(t - T)}, \text{ for } t > T$$

$$= Y, \text{ for } t = T$$

where:

y = present output value for the differentiator
 x = present input to the differentiator
 t = present simulation time
 Y = output value for the differentiator at the last update time
 X = input to the differentiator at the last update time
 T = simulation time at the last update

The value returned by the Differentiator is calculated on demand. The update signal received from the Controller is used only to record the values Y , X , and T used in the calculation.

Table 15-12 Differentiator Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Controller, SequenceNumber, UnitType, InputValue | Keywords for Calculation Objects (see Table 15-2). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 15-13 Differentiator Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>DoubleCalculation</u> | |
| Value | Output inherited from Calculation Object. |

15.6 PIDController



The PIDController object simulates the Proportional-Integral-Differential (PID) type controller that is widely used in electronic control systems.

```
error = SetPoint - x
integral = INTEGRAL + (error - ERROR)*(t - T)
derivative = (error - ERROR)/(t - T)

y' = ProportionalGain/SetPointScale * (error + integral/IntegralTime +
    derivative*DerivativeTime)

y = min( max( y', OutputLow ), OutputHigh )
```

where:

```
y = present output value for the PID controller
x = present ProcessVariable input to the PID controller
t = present simulation time
Y = output value for the PID controller at the last update time
X = ProcessVariable input to the PID controller at the last update time
T = simulation time at the last update
ERROR = value for error at the last update time
INTEGRAL = value for integral at the last update time
```

The value returned by the PIDController is calculated on demand. The update signal received from the Controller is used only to record the values Y, X, T, ERROR, and INTEGRAL used in the calculation.

Table 15-14 PIDController Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Controller, SequenceNumber, UnitType | Keywords for Calculation Objects (see Table 15-2). |
| SetPoint | The set point for the PIDController. The unit type for the set point is given by the UnitType keyword. The set point can be a constant value or any entity that returns a number, such as a calculation object, a TimeSeries, a Probability Distribution, or an Expression. |
| ProcessVariable | The process variable feedback to the PIDController. The unit type for the process variable is given by the UnitType keyword. The process variable can be a constant value or any entity that returns a number, such as a calculation object, a TimeSeries, a Probability Distribution, or an Expression. |
| ProcessVariableScale | A constant with the same unit type as the process variable and the set point. The difference between the process variable and the set point is divided by this quantity to make a dimensionless variable. |
| OutputUnitType | The unit type for the output from the PID controller. |
| ProportionalGain | The coefficient applied to the proportional feedback loop. The unit type for the proportional gain is given by the OutputUnitType keyword. |
| IntegralTime | The coefficient applied to the integral feedback loop. |
| DerivativeTime | The coefficient applied to the differential feedback loop. |
| OutputLow | The lower limit for the output signal. |
| OutputHigh | The upper limit for the output signal. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 15-15 PIDController Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>DoubleCalculation</u> | |
| Value | Output inherited from Calculation Object. |
| <u>PIDController</u> | |
| ProportionalValue | The proportional component of the output value. |
| IntegralValue | The integral component of the output value. |
| DerivativeValue | The derivative component of the output value. |

15.7 Lag



The Lag object calculates the lag operation used in electronic control systems.

$$y = Y + (t - T) * (x - Y) / \text{LagTime}$$

where:

y = present output value for the Lag object
 x = present input to the Lag object
 t = present simulation time
 Y = output value for the Lag object at the last update time
 X = input to the Lag object at the last update time
 T = simulation time at the last update

The value returned by the Lag object is calculated on demand. The update signal received from the Controller is used only to record the values Y , X , and T used in the calculation.

Table 15-16 Lag Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|----------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Controller, SequenceNumber, UnitType, InputValue | Keywords for Calculation Objects (see Table 15-2). |
| LagTime | A value with units of time used in the lag calculations. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 15-17 Lag Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>DoubleCalculation</u> | |
| Value | Output inherited from Calculation Object. |
| <u>Lag</u> | |
| Error | The value for (InputValue - Value). |

15.8 MovingAverage



The MovingAverage object calculates the average value of the input over a given time interval.

$$y = \{ x + \sum_{(i = M \text{ to } M-N-2)} X(i) \} / N$$

where:

y = present output value for the MovingAverage object
 x = present input to the MovingAverage object
 N = **NumberOfSamples**
 M = number of updates that have been performed previously
 $X(i)$ = input to the MovingAverage object at the i^{th} update time

The value returned by the MovingAverage object is calculated on demand. The update signal received from the Controller is used only to record the values $X(i)$ used in the calculation.

Table 15-18 MovingAverage Key Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Controller, SequenceNumber, UnitType, InputValue | Keywords for Calculation Objects (see Table 15-2). |
| NumberOfSamples | The number of input values over which to average. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 15-19 MovingAverage Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>DoubleCalculation</u> | |
| Value | Output inherited from Calculation Object. |

15.9 SineWave



The SineWave object generates a sine wave output:

$$y = \text{Offset} + \text{Amplitude} * \sin(2\pi t / \text{Period} + \text{PhaseAngle})$$

where:

y = present output value for the SineWave object
t = present simulation time

The value returned by the SineWave is calculated on demand.

Table 15-20 SineWave Key Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType | The unit type for the value returned by the wave. |
| Amplitude | Amplitude of the generated wave. |
| Period | Period of the generated wave. |
| PhaseAngle | Initial phase angle of the generated wave. |
| Offset | Offset added to the output of the generated wave. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 15-21 SineWave Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>WaveGenerator</u> | |
| Value | The present value for the wave. |

15.10 SquareWave



The SquareWave object generates a square wave output:

$$y = \text{Offset} + \text{Amplitude}, \text{ if } \sin(2\pi t/\text{Period} + \text{PhaseAngle}) \geq 0$$

$$= \text{Offset} - \text{Amplitude}, \text{ if } \sin(2\pi t/\text{Period} + \text{PhaseAngle}) < 0$$

where:

y = present output value for the SquareWave object
 t = present simulation time

The value returned by the SquareWave is calculated on demand.

Table 15-22 SquareWave Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| UnitType | The unit type for the value returned by the wave. |
| Amplitude | Amplitude of the generated wave. |
| Period | Period of the generated wave. |
| PhaseAngle | Initial phase angle of the generated wave. |
| Offset | Offset added to the output of the generated wave. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 15-23 SquareWave Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>WaveGenerator</u> | |
| Value | The present value for the wave. |

15.11 UnitDelay



The UnitDelay object holds the value from the last update time. It is used to prevent an infinite loop from occurring when a modelled calculation includes a feedback loop.

$$y = x$$

where:

y = present output value for the UnitDelay object
 x = input to the UnitDelay at the last update time

Table 15-24 UnitDelay Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Controller, SequenceNumber, UnitType, InputValue | Keywords for Calculation Objects (see Table 15-2). |
| InitialValue | The value for the UnitDelay function at simulation time equal to zero. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 15-25 UnitDelay Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>UnitDelay</u> | |
| Value | The result of the calculation at the present time. |

16 Fluid Objects Palette

The Fluid Objects Palette contains objects for building dynamic models of hydraulic flow involving tanks, pipes, pumps, etc. The flow rate in the system is calculated by solving the unsteady Bernoulli equation.

The following objects are provided in the Fluid Objects palette.

Table 16-1 Fluid Objects Palette

| Object | Description |
|----------------------|------------------------------------------------------------------------|
| Fluid | Defines a specific fluid and its properties. |
| FluidFlow | Computed flow of a specified fluid between a source and a destination. |
| FluidFixedFlow | Constant flow to/from a specified tank. |
| FluidTank | Cylindrical vessel for storing fluid. |
| FluidPipe | Cylindrical conduit for transporting fluid. |
| FluidCentrifugalPump | Type of pump with a rotating impeller. |

Many fluid objects use the following inputs and outputs.

Table 16-2 Fluid Object Inputs

| Keyword | Description |
|----------|----------------------------------------------------------------------------------------------------------------|
| Previous | The upstream component that feeds this component. |
| Diameter | The hydraulic diameter of the component. Equal to the inside diameter of a pipe with a circular cross-section. |

Table 16-3 Fluid Object Outputs

| Output Name | Description |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------|
| FlowArea | The cross-sectional area of the component. |
| Velocity | The velocity of the fluid within the component. |
| ReynoldsNumber | The Reynolds Number for the fluid within the component. Equal to $(\text{velocity})(\text{diameter})/(\text{kinematic viscosity})$. |
| DynamicPressure | The dynamic pressure of the fluid flow. Equal to $(0.5)(\text{density})(\text{velocity}^2)$. |
| InletPressure | The static pressure at the component's inlet. |
| OutletPressure | The static pressure at the component's outlet. |

16.1 Fluid



The Fluid object defines the basic properties of the fluid.

Table 16-4 Fluid Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Density | The density of the fluid (default = water). |
| Viscosity | The dynamic viscosity of the fluid (default = water). |
| Colour | The colour used to represent the fluid. |
| Gravity | The acceleration of gravity to be used in the fluid flow calculations. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 16-5 Fluid Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |

16.2 FluidFlow



The FluidFlow object calculates the flow rate between a source and a destination by solving the unsteady Bernoulli equation.

Table 16-6 FluidFlow Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Controller, SequenceNumber | Keywords for Calculation Objects (see Table 15-2). |
| Fluid | The Fluid being moved by the flow. |
| Source | The source object for the flow. |
| Destination | The destination object for the flow. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 16-7 FluidFlow Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>FluidFlowCalculation</u> | |
| FlowRate | The volumetric flow rate for the system. |
| <u>FluidFlow</u> | |
| FlowAcceleration | The time derivative of the volumetric flow rate. |
| FlowInertia | The sum of (density)(length)/(flow area) for the hydraulic components in the route. |

16.3 FluidFixedFlow



The FluidFixedFlow object moves Fluid between a source and a destination at a fixed volumetric flow rate.

Table 16-8 FluidFixedFlow Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Controller, SequenceNumber | Keywords for Calculation Objects (see Table 15-2). |
| Fluid | The Fluid being moved by the flow. |
| Source | The source object for the flow. |
| Destination | The destination object for the flow. |
| FlowRate | The constant volumetric flow rate from the source to the destination. |
| Width | The width of the pipe segments in pixels. |
| Colour | The colour of the pipe. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 16-9 FluidFixedFlow Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>FluidFlowCalculation</u> | |
| FlowRate | The volumetric flow rate for the system. |

16.4 FluidTank



The FluidTank object represents a cylindrical storage tank containing Fluid. FluidTanks are modelled as large diameter FluidPipes so that the velocity and inertia of the Fluid are preserved in calculations.

Table 16-10 FluidTank Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Previous, Diameter | Keywords for Fluid Objects (see Table 16-2). |
| Capacity | The total volume of fluid that can be stored in the tank. |
| InitialVolume | The volume of fluid in the tank at the start of the simulation. |
| AmbientPressure | The atmospheric pressure acting on the surface of the fluid in the tank. |
| InletHeight | The height of the flow feeding the tank. Measured relative to the bottom of the tank. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 16-11 FluidTank Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>FluidComponent</u> | |
| FlowArea, Velocity, ReynoldsNumber, DynamicPressure, InletPressure, OutletPressure | Outputs inherited from Fluid Objects (see Table 16-5). |
| <u>FluidTank</u> | |
| FluidVolume | The volume of the fluid stored in the tank. |
| FluidLevel | The height of the fluid from the bottom of the tank. |

16.5 FluidPipe



The FluidPipe object represents a cylindrical pipe for transporting Fluid.

Table 16-12 FluidPipe Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Previous, Diameter | Keywords for Fluid Objects (see Table 16-2). |
| Length | The length of the pipe. |
| HeightChange | The height change over the length of the pipe. Equal to (outlet height - inlet height). |
| Roughness | The roughness height of the inside pipe surface. Used to calculate the Darcy friction factor for the pipe. |
| PressureLossCoefficient | The pressure loss coefficient or 'K-factor' for the pipe. The factor multiplies the dynamic pressure and is applied as a loss at the pipe outlet. |
| Width | The width of the pipe segments in pixels. |
| Colour | The colour of the pipe. |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 16-13 FluidPipe Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>FluidComponent</u> | |
| FlowArea, Velocity, ReynoldsNumber, DynamicPressure, InletPressure, OutletPressure | Outputs inherited from Fluid Objects (see Table 16-5). |
| <u>FluidPipe</u> | |
| DarcyFrictionFactor | The Darcy Friction Factor for the pipe. |

16.6 FluidCentrifugalPump



The FluidCentrifugalPump object models the performance of a centrifugal pump.

Table 16-14 FluidCentrifugalPump Inputs

| Keyword | Description |
|---------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Key Inputs</u> | |
| AttributeDefinitionList, CustomOutputList | Keywords for User-Defined Variables (see Table 6-14). |
| Previous, Diameter | Keywords for Fluid Objects (see Table 16-2). |
| MaxFlowRate | Maximum volumetric flow rate that the pump can generate. |
| MaxPressure | Maximum static pressure that the pump can generate (at zero flow rate). |
| MaxPressureLoss | Maximum static pressure loss for the pump (at maximum flow rate). |
| SpeedController | The CalculationEntity whose output sets the rotational speed of the pump. The output value is ratio of present speed to maximum speed (0.0 - 1.0). |
| <u>Graphics</u> | |
| Position, Alignment, Size, Orientation, Region, RelativeEntity, DisplayModel, Show, Movable | Keywords for Graphics (see Table 10-1). |

Table 16-15 FluidCentrifugalPump Outputs

| Output Name | Description |
|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <u>Entity and DisplayEntity</u> | |
| Name, ObjectType, SimTime, Attributes, Custom Outputs, Position, Size, Orientation, Alignment | Outputs inherited from Entity and DisplayEntity (see Table 10-2). |
| <u>FluidComponent</u> | |
| FlowArea, Velocity, ReynoldsNumber, DynamicPressure, InletPressure, OutletPressure | Outputs inherited from Fluid Objects (see Table 16-5). |

Appendix A - Named Colours

This section provides the names and respective RGB values for the pre-defined colours built into JaamSim.

| Colour Name | Colour | R | G | B |
|-----------------|--------|-----|-----|-----|
| lavenderblush | | 255 | 240 | 245 |
| pink | | 255 | 192 | 203 |
| lightpink | | 255 | 182 | 193 |
| palevioletred | | 219 | 112 | 147 |
| hotpink | | 255 | 105 | 180 |
| deeppink | | 255 | 20 | 147 |
| violetred | | 208 | 32 | 144 |
| mediumvioletred | | 199 | 21 | 133 |
| raspberry | | 135 | 38 | 87 |
| thistle | | 216 | 191 | 216 |
| plum | | 221 | 160 | 221 |
| orchid | | 218 | 112 | 214 |
| violet | | 238 | 130 | 238 |
| magenta | | 255 | 0 | 255 |
| purple | | 128 | 0 | 128 |
| mediumorchid | | 186 | 85 | 211 |
| darkorchid | | 153 | 50 | 204 |
| darkviolet | | 148 | 0 | 211 |
| blueviolet | | 138 | 43 | 226 |
| indigo | | 75 | 0 | 130 |
| mediumpurple | | 147 | 112 | 219 |
| lightslateblue | | 132 | 112 | 255 |
| mediumslateblue | | 123 | 104 | 238 |
| slateblue | | 106 | 90 | 205 |
| darkslateblue | | 72 | 61 | 139 |
| ghostwhite | | 248 | 248 | 255 |
| lavender | | 230 | 230 | 250 |
| blue | | 0 | 0 | 255 |
| darkblue | | 0 | 0 | 139 |
| navy | | 0 | 0 | 128 |
| midnightblue | | 25 | 25 | 112 |
| cobalt | | 61 | 89 | 171 |
| royalblue | | 65 | 105 | 225 |

| Colour Name | Colour | R | G | B |
|-------------|--------|-----|-----|-----|
| chocolate | | 210 | 105 | 30 |
| rawsienna | | 199 | 97 | 20 |
| sienna | | 160 | 82 | 45 |
| brown | | 138 | 54 | 15 |
| lightsalmon | | 255 | 160 | 122 |
| darksalmon | | 233 | 150 | 122 |
| salmon | | 250 | 128 | 114 |
| lightcoral | | 240 | 128 | 128 |
| coral | | 255 | 114 | 86 |
| tomato | | 255 | 99 | 71 |
| orangered | | 255 | 69 | 0 |
| red | | 255 | 0 | 0 |
| crimson | | 220 | 20 | 60 |
| firebrick | | 178 | 34 | 34 |
| indianred | | 176 | 23 | 31 |
| burntumber | | 138 | 51 | 36 |
| maroon | | 128 | 0 | 0 |
| sepia | | 94 | 38 | 18 |
| white | | 255 | 255 | 255 |
| gray99 | | 252 | 252 | 252 |
| gray98 | | 250 | 250 | 250 |
| gray97 | | 247 | 247 | 247 |
| gray96 | | 245 | 245 | 245 |
| gray95 | | 242 | 242 | 242 |
| gray94 | | 240 | 240 | 240 |
| gray93 | | 237 | 237 | 237 |
| gray92 | | 235 | 235 | 235 |
| gray91 | | 232 | 232 | 232 |
| gray90 | | 229 | 229 | 229 |
| gray89 | | 227 | 227 | 227 |
| gray88 | | 224 | 224 | 224 |
| gray87 | | 222 | 222 | 222 |
| gray86 | | 219 | 219 | 219 |

| Colour Name | Colour | R | G | B |
|-------------------|--------|-----|-----|-----|
| cornflowerblue | | 100 | 149 | 237 |
| lightsteelblue | | 176 | 196 | 222 |
| lightslategray | | 119 | 136 | 153 |
| slategray | | 112 | 128 | 144 |
| dodgerblue | | 30 | 144 | 255 |
| aliceblue | | 240 | 248 | 255 |
| powderblue | | 176 | 224 | 230 |
| lightblue | | 173 | 216 | 230 |
| lightskyblue | | 135 | 206 | 250 |
| skyblue | | 135 | 206 | 235 |
| deepskyblue | | 0 | 191 | 255 |
| peacock | | 51 | 161 | 201 |
| steelblue | | 70 | 130 | 180 |
| darkturquoise | | 0 | 206 | 209 |
| cadetblue | | 95 | 158 | 160 |
| azure | | 240 | 255 | 255 |
| lightcyan | | 224 | 255 | 255 |
| paleturquoise | | 187 | 255 | 255 |
| cyan | | 0 | 255 | 255 |
| turquoise | | 64 | 224 | 208 |
| mediumturquoise | | 72 | 209 | 204 |
| lightseagreen | | 32 | 178 | 170 |
| manganeseblue | | 3 | 168 | 158 |
| teal | | 0 | 128 | 128 |
| darkslategray | | 47 | 79 | 79 |
| turquoiseblue | | 0 | 199 | 140 |
| aquamarine | | 127 | 255 | 212 |
| mintcream | | 245 | 255 | 250 |
| mint | | 189 | 252 | 201 |
| seagreen | | 84 | 255 | 159 |
| mediumspringgreen | | 0 | 250 | 154 |
| springgreen | | 0 | 255 | 127 |
| emeraldgreen | | 0 | 201 | 87 |
| mediumseagreen | | 60 | 179 | 113 |
| cobaltgreen | | 61 | 145 | 64 |
| darkseagreen | | 143 | 188 | 143 |
| honeydew | | 240 | 255 | 240 |

| Colour Name | Colour | R | G | B |
|-------------|--------|-----|-----|-----|
| gray85 | | 217 | 217 | 217 |
| gray84 | | 214 | 214 | 214 |
| gray83 | | 212 | 212 | 212 |
| gray82 | | 209 | 209 | 209 |
| gray81 | | 207 | 207 | 207 |
| gray80 | | 204 | 204 | 204 |
| gray79 | | 201 | 201 | 201 |
| gray78 | | 199 | 199 | 199 |
| gray77 | | 196 | 196 | 196 |
| gray76 | | 194 | 194 | 194 |
| gray75 | | 191 | 191 | 191 |
| gray74 | | 189 | 189 | 189 |
| gray73 | | 186 | 186 | 186 |
| gray72 | | 184 | 184 | 184 |
| gray71 | | 181 | 181 | 181 |
| gray70 | | 179 | 179 | 179 |
| gray69 | | 176 | 176 | 176 |
| gray68 | | 173 | 173 | 173 |
| gray67 | | 171 | 171 | 171 |
| gray66 | | 168 | 168 | 168 |
| gray65 | | 166 | 166 | 166 |
| gray64 | | 163 | 163 | 163 |
| gray63 | | 161 | 161 | 161 |
| gray62 | | 158 | 158 | 158 |
| gray61 | | 156 | 156 | 156 |
| gray60 | | 153 | 153 | 153 |
| gray59 | | 150 | 150 | 150 |
| gray58 | | 148 | 148 | 148 |
| gray57 | | 145 | 145 | 145 |
| gray56 | | 143 | 143 | 143 |
| gray55 | | 140 | 140 | 140 |
| gray54 | | 138 | 138 | 138 |
| gray53 | | 135 | 135 | 135 |
| gray52 | | 133 | 133 | 133 |
| gray51 | | 130 | 130 | 130 |
| gray50 | | 127 | 127 | 127 |
| gray49 | | 125 | 125 | 125 |

| Colour Name | Colour | R | G | B |
|----------------------|--------|-----|-----|-----|
| palegreen | | 152 | 251 | 152 |
| lawngreen | | 124 | 252 | 0 |
| greenyellow | | 173 | 255 | 47 |
| limegreen | | 50 | 205 | 50 |
| forestgreen | | 34 | 139 | 34 |
| sapgreen | | 48 | 128 | 20 |
| green | | 0 | 128 | 0 |
| darkgreen | | 0 | 100 | 0 |
| darkolivegreen | | 85 | 107 | 47 |
| olivedrab | | 107 | 142 | 35 |
| olive | | 128 | 128 | 0 |
| ivory | | 255 | 255 | 240 |
| lightyellow | | 255 | 255 | 224 |
| lightgoldenrodyellow | | 250 | 250 | 210 |
| cornsilk | | 255 | 248 | 220 |
| lemonchiffon | | 255 | 250 | 205 |
| beige | | 245 | 245 | 220 |
| yellow | | 255 | 255 | 0 |
| khaki | | 240 | 230 | 140 |
| lightgoldenrod | | 255 | 236 | 139 |
| palegoldenrod | | 238 | 232 | 170 |
| darkkhaki | | 189 | 183 | 107 |
| banana | | 227 | 207 | 87 |
| gold | | 255 | 215 | 0 |
| goldenrod | | 218 | 165 | 32 |
| darkgoldenrod | | 184 | 134 | 11 |
| brick | | 156 | 102 | 31 |
| floralwhite | | 255 | 250 | 240 |
| seashell | | 255 | 245 | 238 |
| oldlace | | 253 | 245 | 230 |
| linen | | 250 | 240 | 230 |
| antiquewhite | | 250 | 235 | 215 |
| papayawhip | | 255 | 239 | 213 |
| blanchedalmond | | 255 | 235 | 205 |
| eggshell | | 252 | 230 | 201 |
| bisque | | 255 | 228 | 196 |
| moccasin | | 255 | 228 | 181 |

| Colour Name | Colour | R | G | B |
|-------------|--------|-----|-----|-----|
| gray48 | | 122 | 122 | 122 |
| gray47 | | 120 | 120 | 120 |
| gray46 | | 117 | 117 | 117 |
| gray45 | | 115 | 115 | 115 |
| gray44 | | 112 | 112 | 112 |
| gray43 | | 110 | 110 | 110 |
| gray42 | | 107 | 107 | 107 |
| gray41 | | 105 | 105 | 105 |
| gray40 | | 102 | 102 | 102 |
| gray39 | | 99 | 99 | 99 |
| gray38 | | 97 | 97 | 97 |
| gray37 | | 94 | 94 | 94 |
| gray36 | | 92 | 92 | 92 |
| gray35 | | 89 | 89 | 89 |
| gray34 | | 87 | 87 | 87 |
| gray33 | | 84 | 84 | 84 |
| gray32 | | 82 | 82 | 82 |
| gray31 | | 79 | 79 | 79 |
| gray30 | | 77 | 77 | 77 |
| gray29 | | 74 | 74 | 74 |
| gray28 | | 71 | 71 | 71 |
| gray27 | | 69 | 69 | 69 |
| gray26 | | 66 | 66 | 66 |
| gray25 | | 64 | 64 | 64 |
| gray24 | | 61 | 61 | 61 |
| gray23 | | 59 | 59 | 59 |
| gray22 | | 56 | 56 | 56 |
| gray21 | | 54 | 54 | 54 |
| gray20 | | 51 | 51 | 51 |
| gray19 | | 48 | 48 | 48 |
| gray18 | | 46 | 46 | 46 |
| gray17 | | 43 | 43 | 43 |
| gray16 | | 41 | 41 | 41 |
| gray15 | | 38 | 38 | 38 |
| gray14 | | 36 | 36 | 36 |
| gray13 | | 33 | 33 | 33 |
| gray12 | | 31 | 31 | 31 |

| Colour Name | Colour | R | G | B |
|---------------|--------|-----|-----|-----|
| navajowhite | | 255 | 222 | 173 |
| wheat | | 245 | 222 | 179 |
| peachpuff | | 255 | 218 | 185 |
| tan | | 210 | 180 | 140 |
| burlywood | | 222 | 184 | 135 |
| melon | | 227 | 168 | 105 |
| sandybrown | | 244 | 164 | 96 |
| cadmiumyellow | | 255 | 153 | 18 |
| carrot | | 237 | 145 | 33 |
| orange | | 255 | 128 | 0 |
| flesh | | 255 | 125 | 64 |
| cadmiumorange | | 255 | 97 | 3 |

| Colour Name | Colour | R | G | B |
|-------------|--------|----|----|----|
| gray11 | | 28 | 28 | 28 |
| gray10 | | 26 | 26 | 26 |
| gray9 | | 23 | 23 | 23 |
| gray8 | | 20 | 20 | 20 |
| gray7 | | 18 | 18 | 18 |
| gray6 | | 15 | 15 | 15 |
| gray5 | | 13 | 13 | 13 |
| gray4 | | 10 | 10 | 10 |
| gray3 | | 8 | 8 | 8 |
| gray2 | | 5 | 5 | 5 |
| gray1 | | 3 | 3 | 3 |
| black | | 0 | 0 | 0 |