Table of Contents

- Erase
- Set Color To
- Make
- Broadcast
- Wait
- Play Sound
- Say
- Set
- Map
- Plot Attribute
- Open URL
- Load Background
- Set Direction
- Stop Simulation
- Reset Simulation
- Clear Simulation

Triggers

- While Running
- On
- Tool
- When Creating A New Agent

Parameters

- Attribute

Language Reference Manual

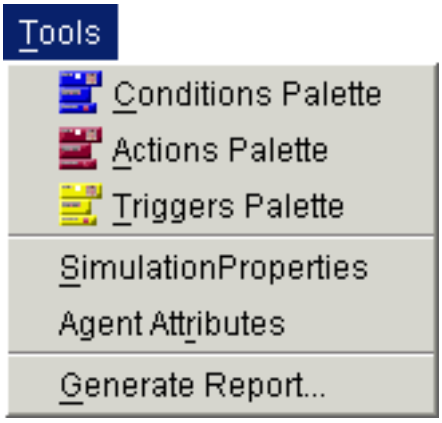
AgentSheets 2.1

Introduction

AgentSheets 2.1 for Windows is an agent-based Web authoring tool enabling a wide range of end users (from children to professionals) to create their own interactive simulations, domain-oriented visual programming languages, and games.

The **Language Reference Manual** provides important information about **Visual AgenTalk** commands and parameters. Use the Language Reference Manual along with the **Getting Started** manual and the complete **Reference Manual** to build your own projects and simulations.

Tools Menu



Description: The **Tools Menu** is used to access the palettes containing Visual AgenTalk language pieces such as conditions, actions and triggers. It is also used to access the simulation properties editor and agent attributes editors. These options are only enabled when an AgentSheets project is open.

The Tools Menu includes the following options:

Conditions Palette
Actions Palette
Triggers Palette
Simulation Properties
Agent Attributes

Conditions

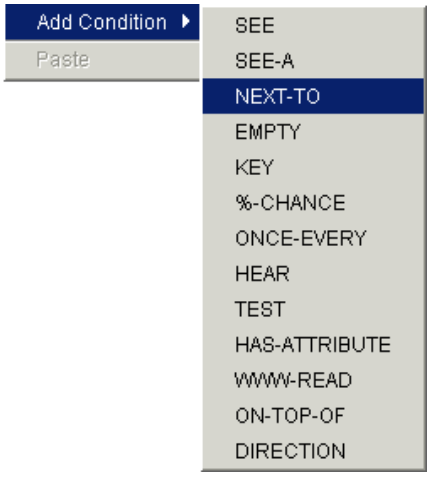
Definition: Conditions are used to test the circumstances agents are in. Among other things, agents can test for the presence of other agents in the worksheet, attribute values, keyboard input, mouse events and even content of live web pages.

Use the **Tools | Conditions Palette** menu option or click the **Conditions** tool in the **Toolbar** to access the Conditions Palette, which contains the following conditions:

See
See A
Next To
Empty
Key
% Chance
Once Every
Hear
Test
Has Attribute
WWW Read
On Top Of
Direction

Power User Shortcuts

A condition can also be accessed and added to the agent's behavior by right-clicking in the part of the rule where conditions reside. This will show the following pop-up menu, from which you can select the **Add Condition** menu option and then select the desired condition to add. The example below illustrates the addition of the **Next To** condition.



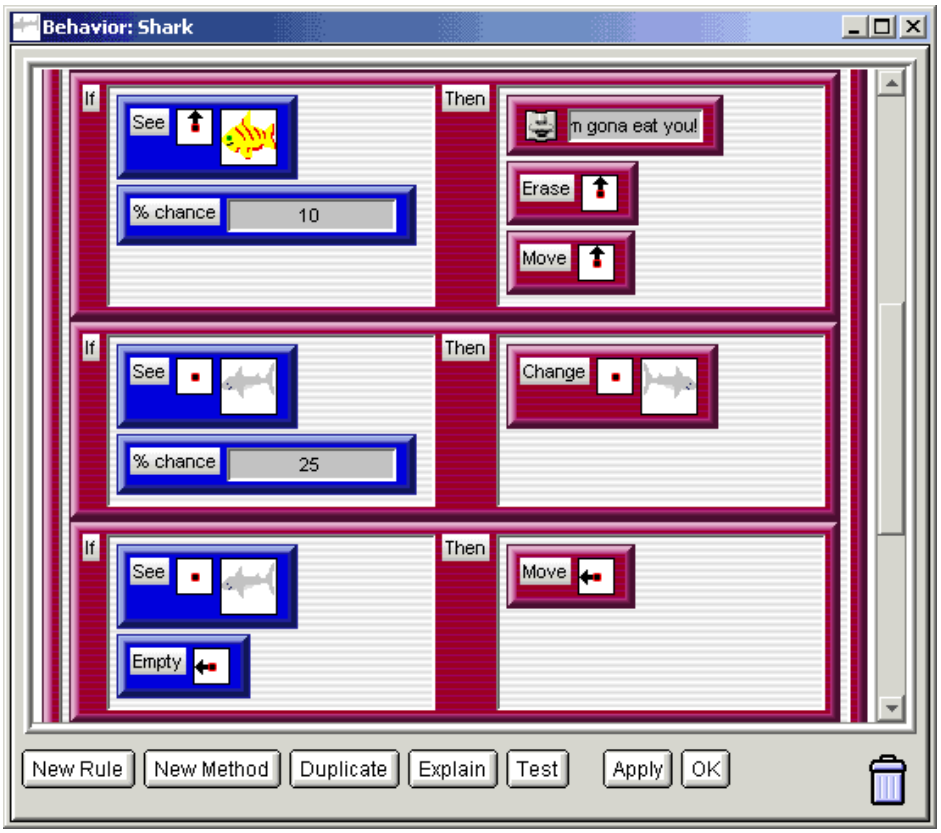
See Condition

Definition: The **See** condition checks if the agent at the cell indicated by the **Direction** parameter looks like the depiction specified in the **Depiction** parameter. The dot (.) refers to the agent itself.

If the indicated cell contains the chosen depiction, the condition is true. Otherwise, the condition is false.

Parameters: **direction**, **depiction**

Example: The Shark agent in the **Fish Tank** simulation looks above it to see if there is a Fish agent to eat, and also checks to see if the Shark itself looks like a Shark going left and decides to change directions or move to the left.



See the Fish Tank simulation on the AgentSheets CD-ROM.

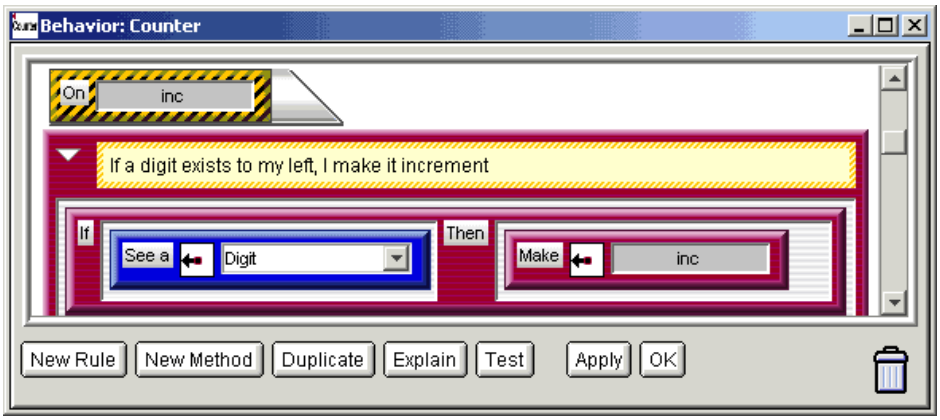
See A Condition

Definition: The **See A** condition checks if the agent at the cell indicated by the **Direction** parameter is of the **Class** specified in the **Class** parameter. This command picks out all the depictions of a given agent class. Use it to identify an agent regardless of the way it looks.

If the indicated cell contains an agent of that class, the condition is true. Otherwise, the condition is false.

Parameters: **direction**, **class**

Example: The Counter agent in the **Benchmark** project checks to see if there is a Digit agent to its left and sends the message to "inc" (increment).



See the Benchmark simulation on the AgentSheets CD-ROM.

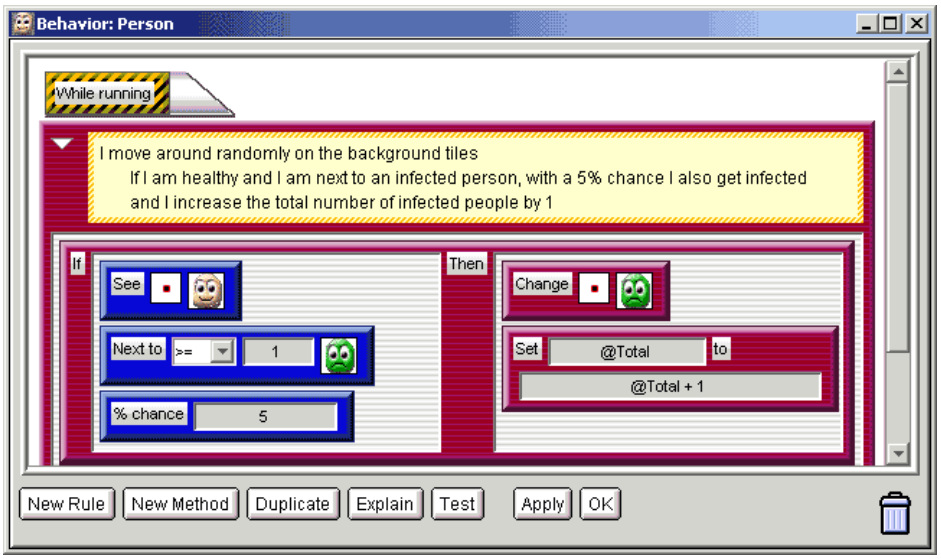
Next To Condition

Definition: The **Next To** condition checks to see how many of the adjacent neighbors have a certain **Depiction**.

If the executing agent is next to the right number of the specified depictions, the condition is true. Otherwise, the condition is false.

Parameters: **comparator**, **formula**, **depiction**

Example: The Person agent in the **Virus Attack** project checks if it is next to one or more sick Person agents. If so, it has a 5% chance of acquiring the virus.



See the Virus Attack simulation on the AgentSheets CD-ROM.

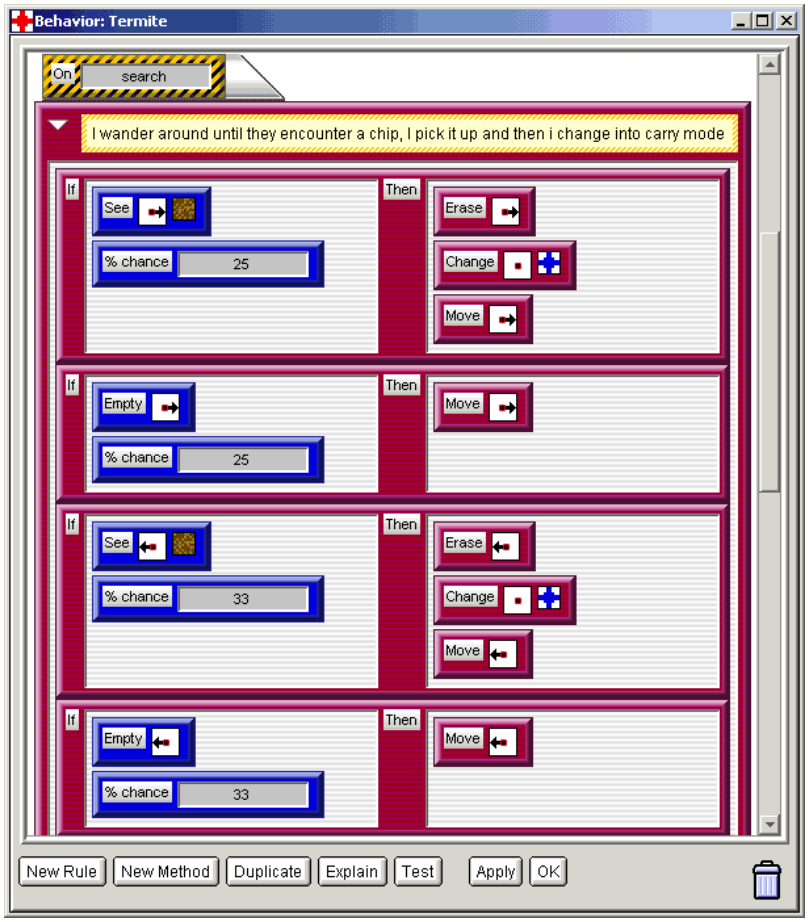
Empty Condition

Definition: The **Empty** condition checks if the neighboring cell specified by the **Direction** parameter is empty. A cell is empty if it does not contain any agents found in the current Gallery.

If the cell indicated by the **Direction** parameter is empty, the condition is true. Otherwise, the condition is false.

Parameters: **direction**

Example: In its search for wood, the Termite agent in the **Termites** project checks if the area to the right or left is empty and has a certain chance (percentage) of moving to that empty position.



See the Termites simulation on the AgentSheets CD-ROM.

Key Condition

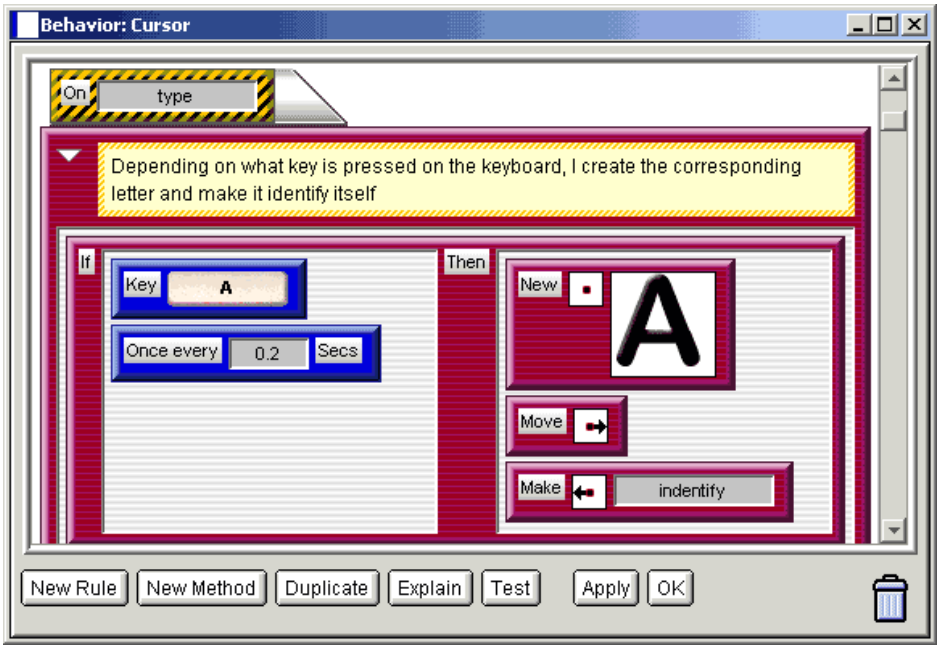
Definition: The **Key** condition checks if the key displayed in its **Key** parameter is pressed.

If the key indicated is pressed, the condition is true. Otherwise, the condition is false.

The **Key** condition can be used to assign real-time control characters on the keyboard. You might also use it to assign "hot key" values to get agents to act out specified actions. To get **Key** conditions to function as hot keys you will have to place them in rules in a method that is checked at every simulation cycle.

Parameters: **key**

Example: The Cursor agent in the **ABC** project detects key strokes and types the corresponding letter. For example, if the **A** key is pressed on the keyboard, the Cursor will create a Character **A** agent, advance to the right, and identify the newly created character.



See the ABC simulation on the AgentSheets CD-ROM.

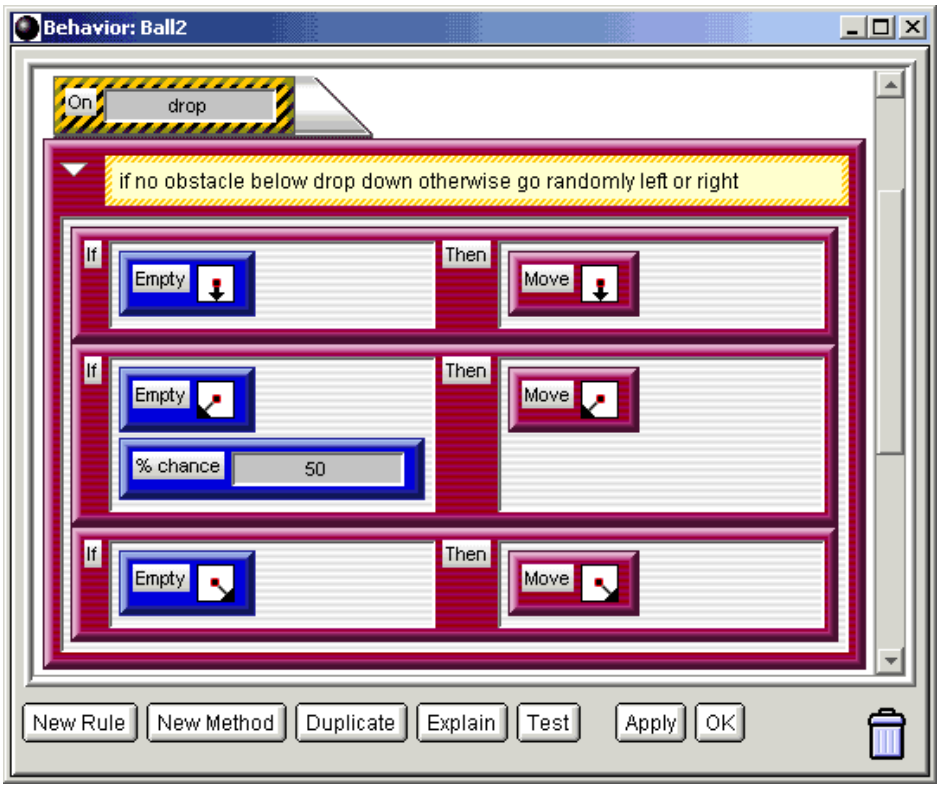
% Chance Condition

Definition: The **% Chance** condition succeeds with a certain percent chance, as specified by a number or a Visual AgenTalk **Formula**. For instance, with a value of 50 this condition will succeed, on average, 50% of the time.

The **% Chance** condition chooses a random number between 1 and 100 whenever it is executed. If this number is less than the number indicated in its number field, the condition is true. Otherwise, the condition is false.

Parameters: **formula**

Example: The Ball agent in the **Counting Pachinko** project moves down if it does not encounter any obstacle, but when it moves down it also has a 50% chance of going to the left or to the right.



See the Counting Pachinko simulation on the AgentSheets CD-ROM.

Once Every Condition

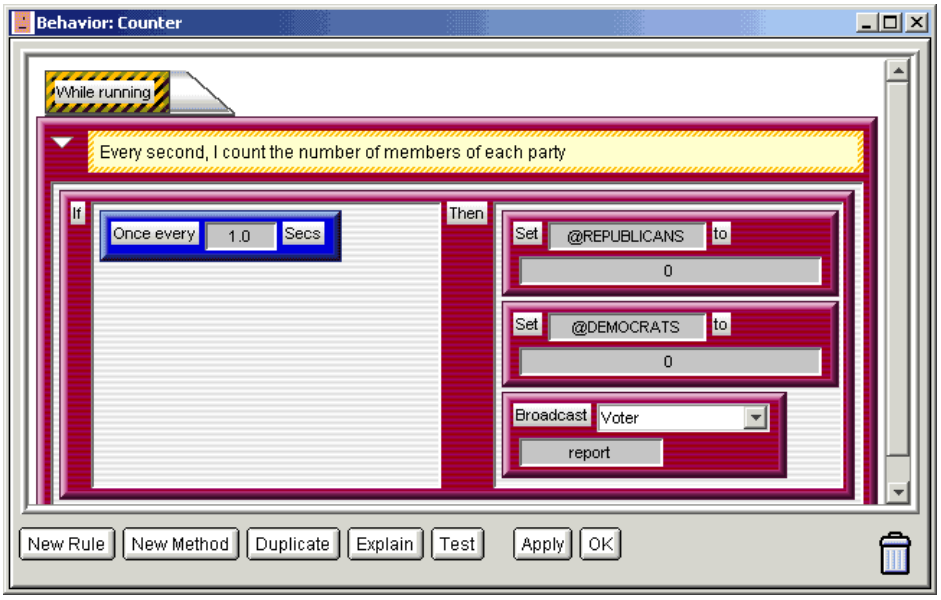
Definition: The **Once Every** condition checks if *N* seconds have passed since it was last checked. *N* is specified by a number or a Visual AgenTalk **Formula**.

If *N* seconds have passed, the condition is true. Otherwise, it is false.

The **Once Every** condition acts as a local timer for an agent. Unlike the **Wait** action, **Once Every** can be used to slow down individual agents without stopping other agents.

Parameters: **formula**

Example: In the **Voting Game** project, the Counter agent polls the voters once every second.



See the Voting Game simulation on the AgentSheets CD-ROM.

Hear Condition

Definition: The **Hear** condition checks the agents in its immediate eight-cell neighborhood to see if any of them is "speaking" the specified text using the **Say** action.

If at least one neighbor is "speaking" the specified text, then this condition is true. Otherwise, it is false.

Parameters: **string**

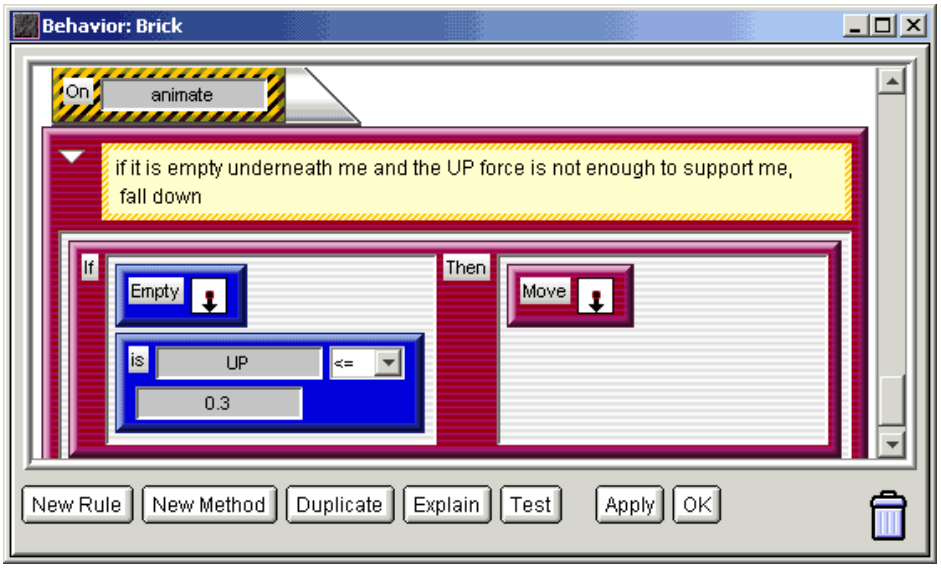
Test Condition

Definition: The **Test** condition compares the values of two **formulae**.

If the comparison yields true, then the condition is true. Otherwise, it is false.

Parameters: **comparator**, **formula**

Example: In the **Bridge Builder** project, the Brick drops down if the cell below it is empty, and the force acting on it is less than or equal to 0.3 (and therefore cannot hold it in place).



See the Bridge Builder simulation on the AgentSheets CD-ROM.

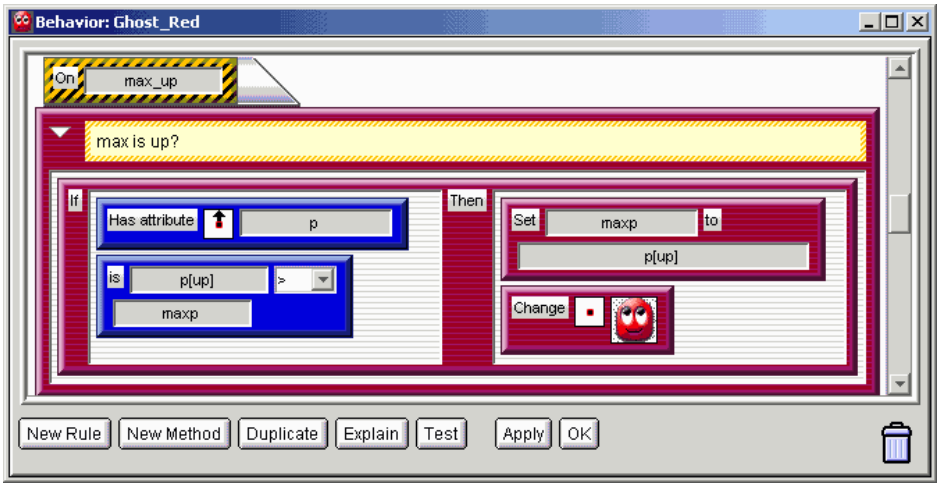
Has Attribute Condition

Definition: The **Has Attribute** condition checks if the agent in the direction specified by the **Direction** parameter has the **Attribute** named in the condition's text field.

If the specified agent has the **Attribute** named, the condition is true. Otherwise, it is false.

Parameters: **direction**, **attribute**

Example: In the **Ultimate Pacman** project, the Red Ghost agent checks if the agent above it has the attribute *p*. It then compares this attribute *p* to the current maxp (the maximum p) and, if *p* is greater, sets maxp to *p*. The Red Ghost then changes its facial expression to look as if it is going upwards. This is part of the behavior for tracking down the Pacman agent (see the Ultimate Pacman "Readme" file for a complete explanation of the tracking behavior).



See the Ultimate Pacman simulation on the AgentSheets CD-ROM.

WWW Read Condition

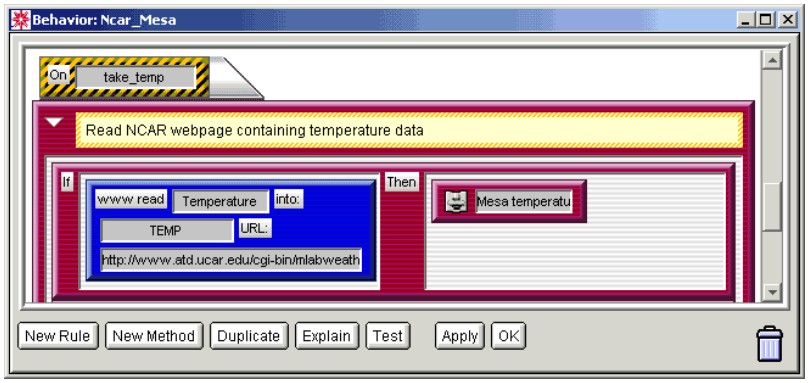
Definition: The **WWW Read** command searches an entire web page specified by the URL in the **URL** parameter for the **String** specified in the first text field.

If the string is found in the web page, a number following that string is read and stored in the attribute specified by the **Attribute** parameter. The search string can contain wildcards (e.g., "*").

This command should only be used when you already have a live connection to the Internet. If you use this command without a live connection the AgentSheets environment will try to open a connection for you.

Parameters: **string**, **attribute**, **URL**

Example: In the **Boulder Live** project, any of the location agents (such as the NCAR_Mesa agent) takes the current temperature by going to the http://www.atd.ucar.edu/cgi-bin/mlabweatherE web page, finding the string "Temperature," and storing it into the **temp** variable, then announces the temperature read from the specified web page.



See the Boulder Live simulation on the AgentSheets CD-ROM.

Ristretto Information: Due to the Java applet security restrictions in some browsers, the **WWW Read** condition may not work in Ristretto-generated applets.

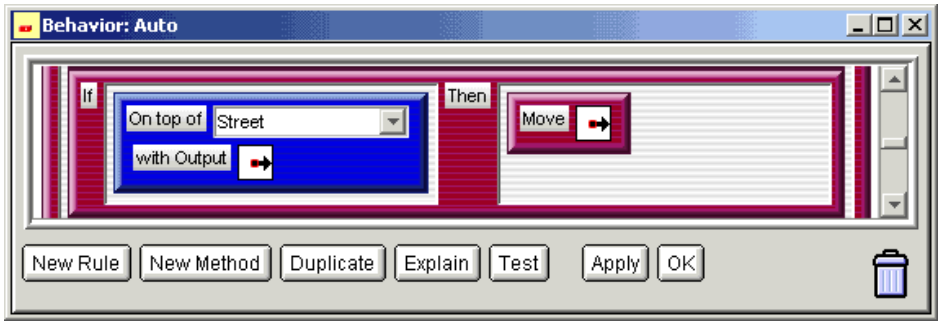
On Top Of Condition

Definition: The **On Top Of** condition checks whether the agent is on top of a specified type of agent with output connectivity in the specified direction.

If the agent below has output connectivity in the direction indicated by the **Direction** parameter, the condition is true. Otherwise, the condition is false.

Parameters: **class**, **direction**

Example: In the **Sustainopolis** project, the Auto agent checks to see if it is on top of a piece of road whose output is right. If it is, the Auto agent moves right.



See the Sustainopolis simulation on the AgentSheets CD-ROM.

Direction Condition

Definition: The **Direction** condition checks if the agent is headed in the direction indicated by the **Direction** parameter.

If the direction of the executing agent is the same as the direction indicated by the **Direction** parameter, then the condition is true. Otherwise, it is false.

Parameters: **direction**

Actions

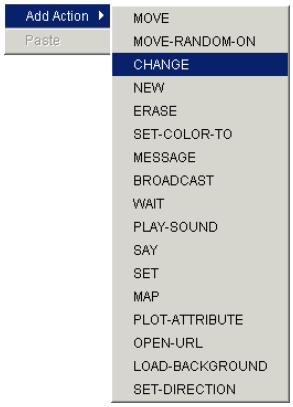
Definition: **Actions** are operations performable by agents. **Actions** allow agents to do things such as moving around in worksheets, changing their depiction, playing a sound, or opening up a web page.

Use the **Tools** | **Actions Palette** menu option or click the **Actions** tool in the **Toolbar** to access the Actions Palette, which contains the following actions:

Move
Move Random On
Change
New
Erase
Set Color To
Make
Broadcast
Wait
Play Sound
Say
Set
Map
Plot Attribute
Open URL
Load Background
Set Direction
Stop Simulation
Reset Simulation
Clear Simulation

Power User Shortcuts

An action can also be accessed and added to the agent's behavior by right-clicking in the part of the rule where actions reside. This will show the following pop-up menu, from which you can select the **Add Action** menu option and then select the desired action to add. The example below illustrates the addition of the **Change action**.


Add Action
Paste
MOVE
MOVE-RANDOM-ON
CHANGE
NEW
ERASE
SET-COLOR-TO
MESSAGE
BROADCAST
WAIT
PLAY-SOUND
SAY
SET
MAP
PLOT-ATTRIBUTE
OPEN-URL
LOAD-BACKGROUND
SET-DIRECTION

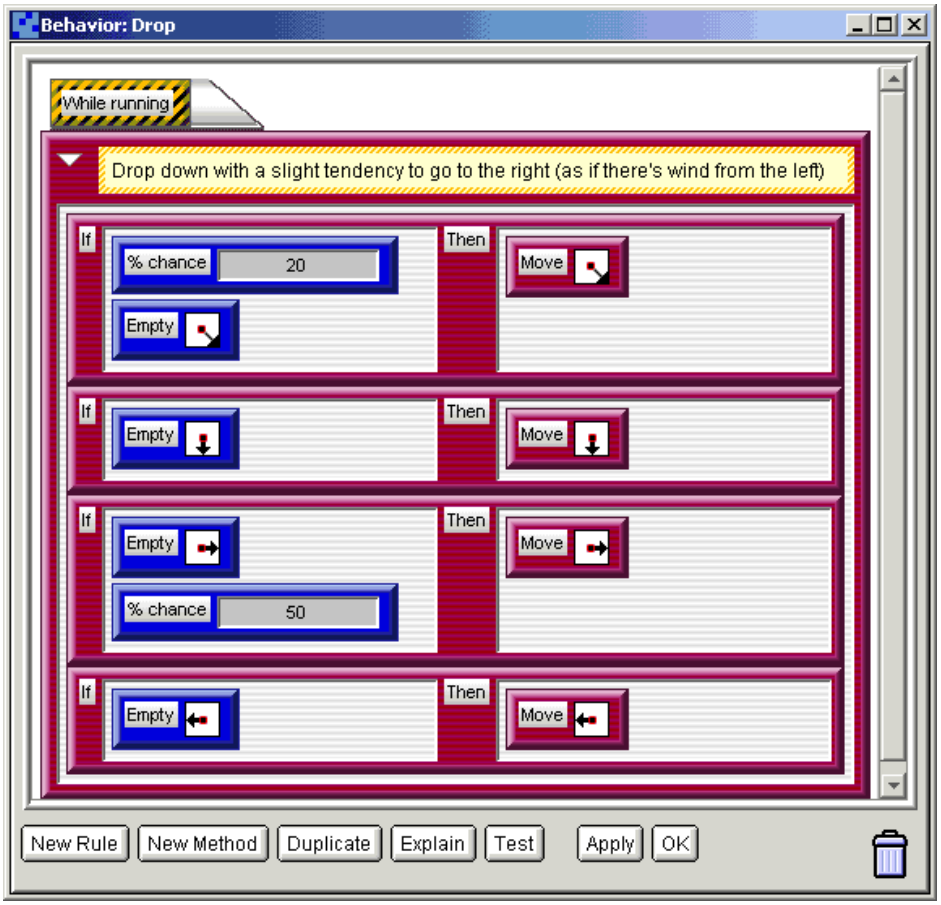
Move Action

Definition: The **Move** action moves an agent one cell in the direction indicated by the **Direction** parameter. The dot (.) will leave the agent at its current position.

Parameters: **direction**

Example: In the **Rainy Day** simulation, the Rain agent drops down with a slight chance of moving diagonally to simulate wind coming from the left.

When the Rain agent can move down no further, it moves left or right.



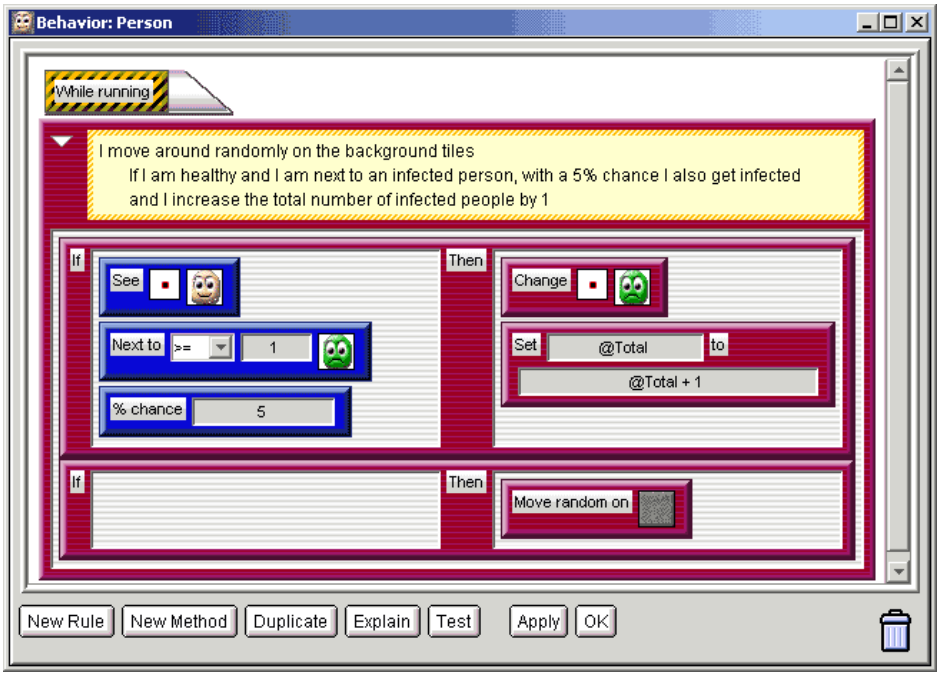
See the Rainy Day simulation on the AgentSheets CD-ROM.

Move Random On Action

Definition: The **Move Random On** action lets an agent move randomly onto any of its immediate neighbors that show the **Depiction** displayed in the **Depiction** parameter. The agent chooses one of its qualifying neighbors to move onto at random.

Parameters: **depiction**

Example: In the **Virus Attack** simulation, the Person agent moves randomly on the background tiles.



See the Virus Attack simulation on the AgentSheets CD-ROM.

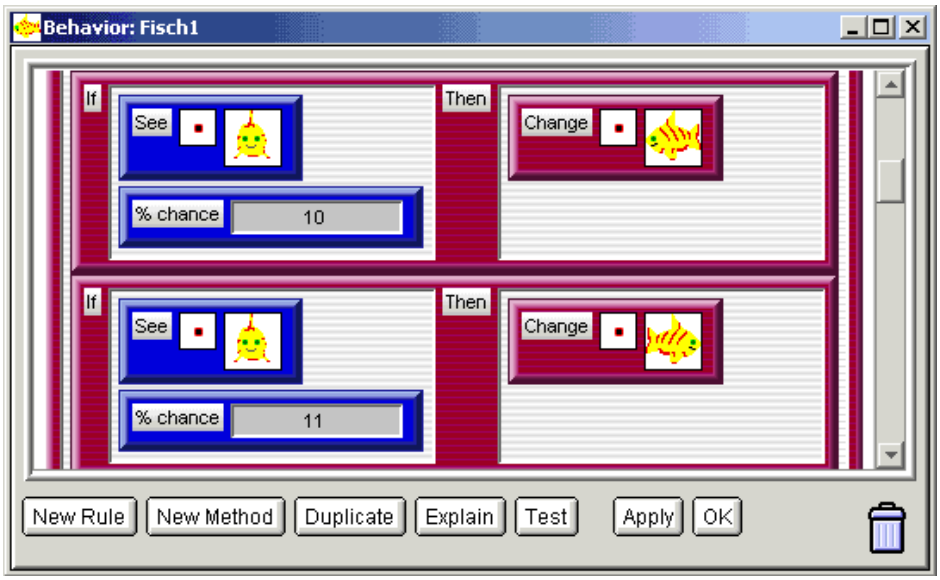
Change Action

Definition: The **Change** action changes the depiction of the agent in the cell indicated by the **Direction** parameter to the depiction displayed in the command's **Depiction** parameter. The dot (.) refers to the agent itself.

The **Change** action is useful for creating animations.

Parameters: **direction**, **depiction**

Example: In the **Fish Tank** simulation, whenever the Fish agent faces forward, there is a 10% chance that it will change to look like the fish going to the left and an 11% chance that it will change to look like the fish going to right.



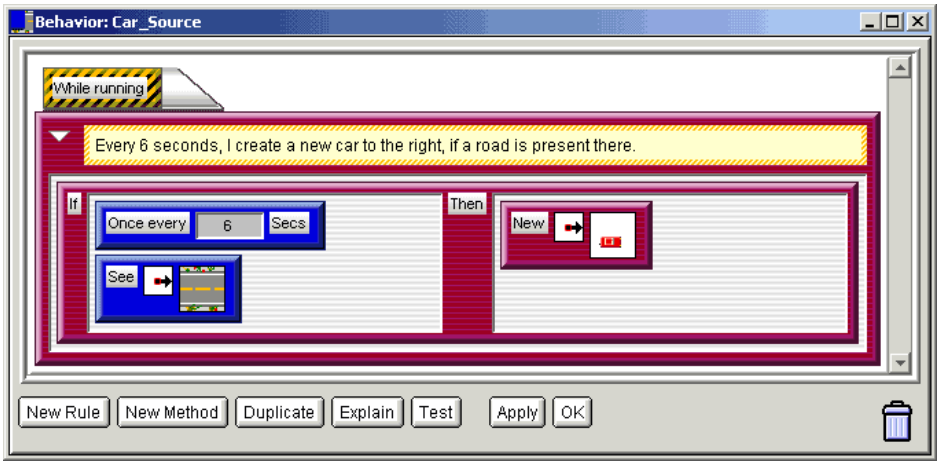
See the Fish Tank simulation on the AgentSheets CD-ROM.

New Action

Definition: The **New** action makes a new agent in the cell indicated by the **Direction** parameter. The new agent has the depiction displayed in the action's **Depiction** parameter.

Parameters: **depiction**, **direction**

Example: The Car-Source agent in the **Sustainopolis** simulation creates new cars on the roads once every six seconds.



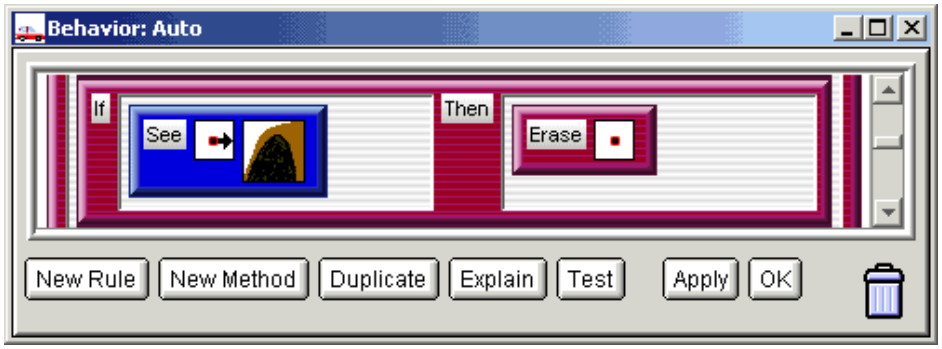
See the Sustainopolis simulation on the AgentSheets CD-ROM.

Erase Action

Definition: The **Erase** action erases the agent located in the cell indicated by the **Direction** parameter. Executing the **Erase** action is the same as applying the **Eraser** tool.

Parameters: **direction**

Example: The Car agent in the **Bridge Builder** simulation erases itself if it sees a tunnel to its right, so it will look like it is going into the tunnel.



See the Bridge Builder simulation on the AgentSheets CD-ROM.

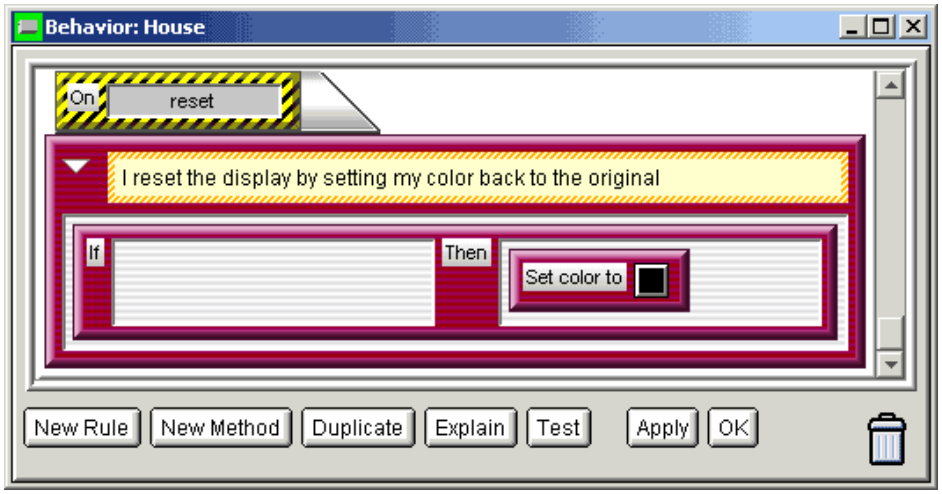
Set Color To Action

Definition: The **Set Color To** action colorizes the agent with the color indicated in the action's **Color** parameter.

The **Color** parameter on the action expands into a full **Color Palette** when you click and hold the mouse on it.

Parameters: **color**

Example: The House agent in the **Sustainopolis** simulation resets to its original color when it receives the "reset" message.



See the Sustainopolis simulation on the AgentSheets CD-ROM.

Make Action

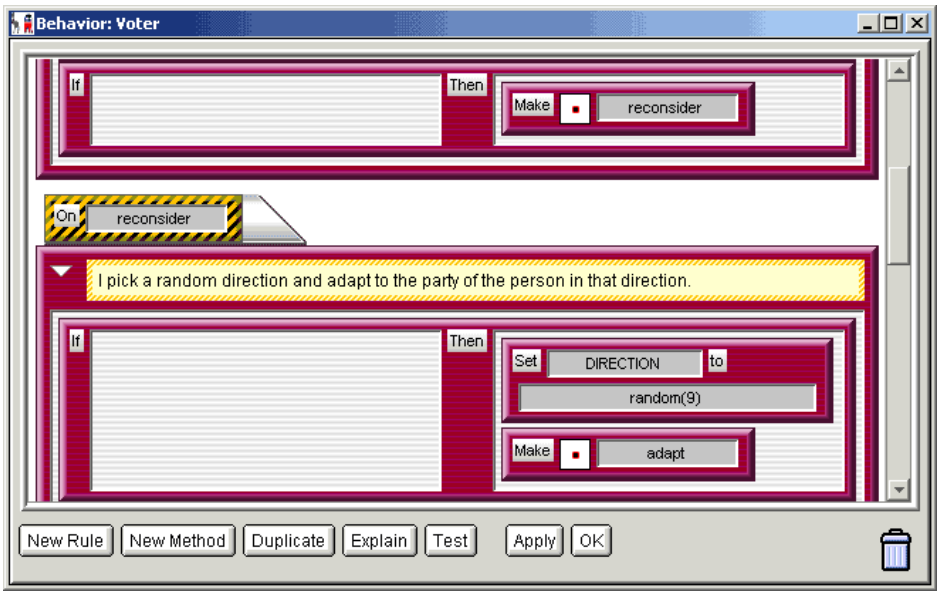
Definition: The **Make** action sends the message specified in the text field to the agent in the cell indicated by the **Direction** parameter. The dot (.) refers to the agent itself; in this case, the agent sends a message to itself.

The **Make** action is used in conjunction with **On** triggers; the receiver can react to the message via the **On** trigger. These triggers give **names** to groups of rules (methods) that can be called using the **Make** action.

By using **Make** actions and **On** triggers, a VAT programmer can divide the functionality of an agent in more meaningful chunks.

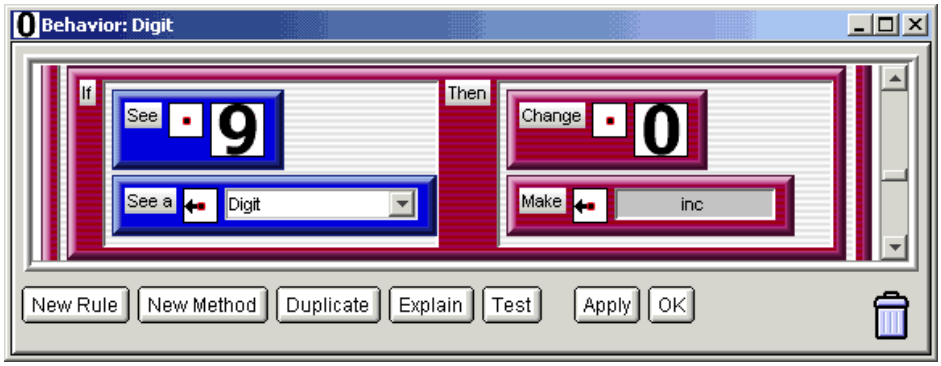
Parameters: **direction**, **method name**

Example 1: In the **Voting Game** simulation, the Voter agent sends a message to itself. It makes itself reconsider its political views by choosing a random direction around it and making itself adapt to the political views of the voter in that direction.



See the Voting Game simulation on the AgentSheets CD-ROM.

Example 2: In the **Benchmark** simulation, the Digit agent sends a message to another agent. When the Digit agent reaches 9, it changes itself to 0 and sends a message to the adjacent digit to the left to increment itself.



See the Benchmark simulation on the AgentSheets CD-ROM.

Broadcast Action

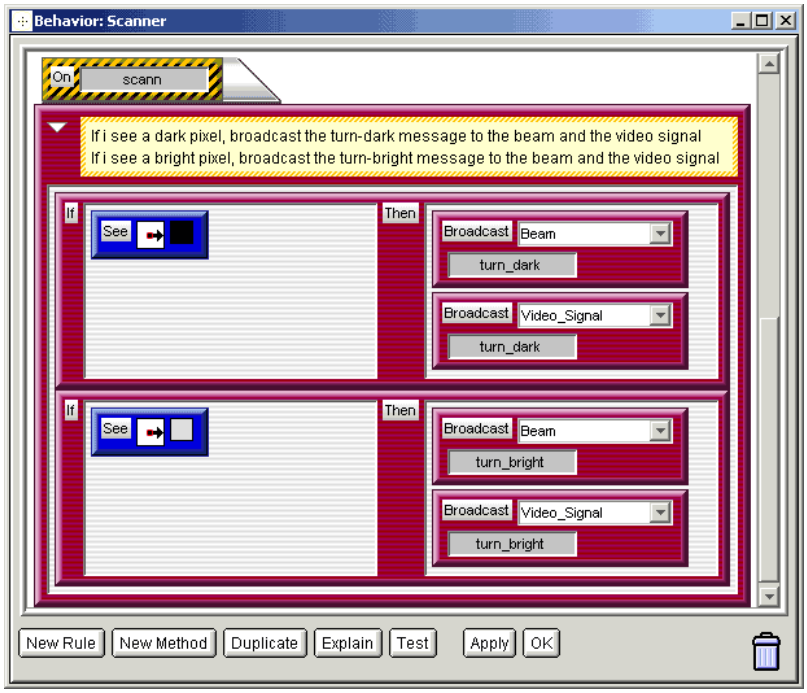
Definition: The **Broadcast** action broadcasts a message to all agents of the class indicated in the action's **Class** parameter.

The message to be broadcast is indicated in the action's text field and is a method call to all agents of the indicated **Class**.

The **Broadcast** action is used in conjunction with **On** triggers; the receivers can react to the message via the **On** trigger. These triggers give **names** to groups of rules (methods) that can be called using the **Broadcast**action.

Parameters: **class**, **method name**

Example: The Scanner agent in the **How Does TV Work?** simulation broadcasts to the TV Beam and Video-Signal agents the message to turn bright or dark, depending on what it scans.



See the How Does TV Work simulation on the AgentSheets CD-ROM.

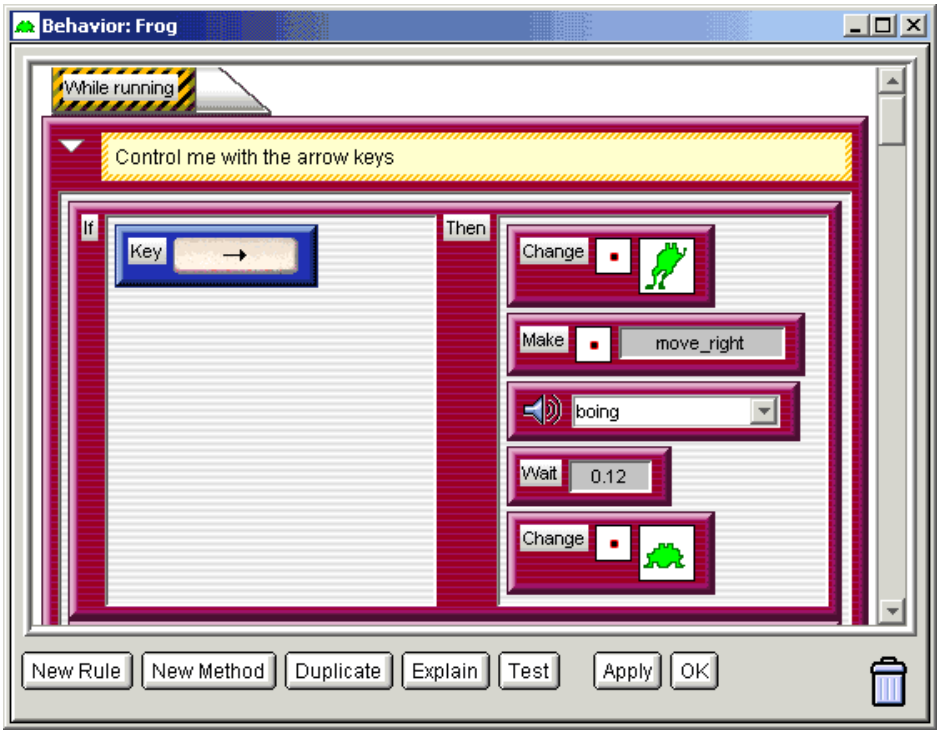
Wait Action

Definition: The **Wait** action causes the **AgentSheets** scheduler to wait the amount of seconds indicated by a number or a Visual AgenTalk **Formula** before executing its next command. Use it to slow down simulations.

Note: The **Wait** action halts the entire simulation. Another alternative is to use the **Once Every** condition to selectively slow down portions of a simulation.

Parameters: **formula**

Example: The Frog agent in the **Frogger** simulation has some delays when controlled with the arrow keys so that it does not move too fast.



See the Frogger simulation on the AgentSheets CD-ROM.

Play Sound Action

Definition: The **Play Sound** action plays the sound chosen in its **Sound** parameter.

Parameters: **sound**

Example: The Car agent in the **Bridge Builder** simulation plays an explosion sound when it drops from the bridge.

See the Bridge Builder simulation on the AgentSheets CD-ROM.

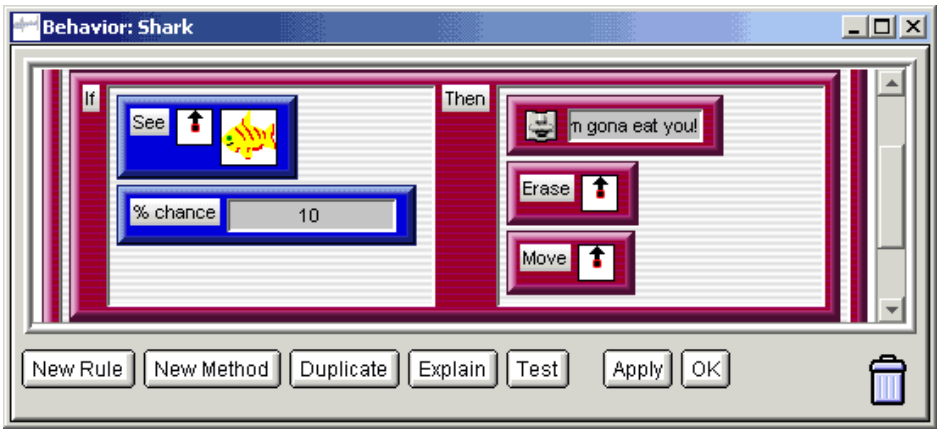
Say Action

Definition: The **Say** action prints the text located in its **String** parameter in the worksheet's status bar.

Unlike its Macintosh version equivalent, the Say action does not yet speak the specified text, since there is no built-in speech synthesizer in Java.

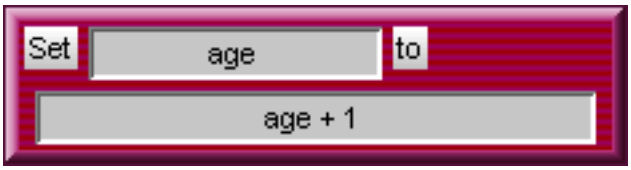
Parameters: **string**

Example: The Shark agent in the **Fish Tank** simulation talks to the Fish agent before eating it.



See the Fish Tank simulation on the AgentSheets CD-ROM.

Set Action



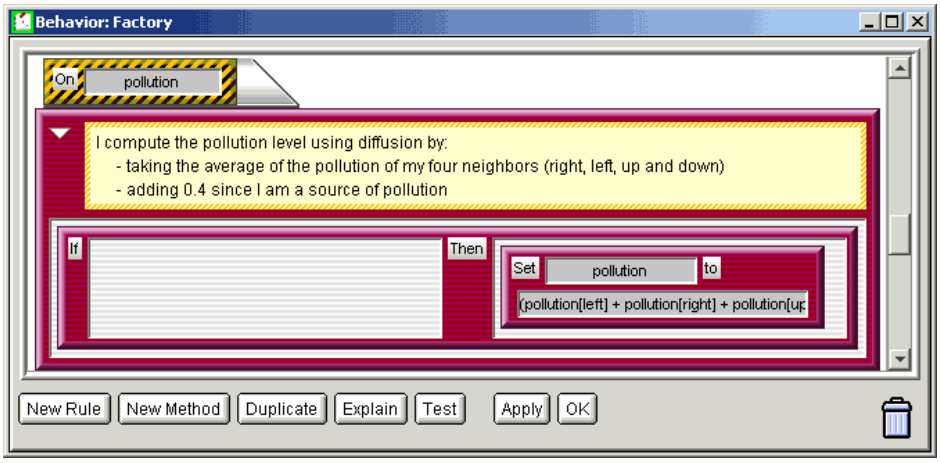
Definition: The **Set** action sets the value of an **Attribute** of the agent or the value of a **Simulation Property** to the computed value of a **Formula**.

The attribute or the simulation property to be set is displayed in the action's text field. The value to set the attribute to is displayed in the action's **Formula** parameter.

Parameters: **attribute**, **formula**

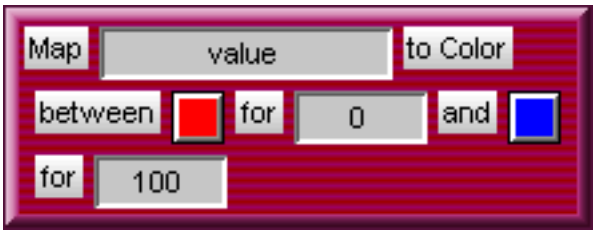
Example: The Factory agent in the **Sustainopolis** simulation computes pollution as the average of the pollution of the things around it, plus an additional amount which represents the pollution from the factory itself.

The formula used is [(pollution[left] + pollution[right] + pollution[up] + pollution[down])/4.0 + 0.4



See the Sustainopolis simulation on the AgentSheets CD-ROM.

Map Action

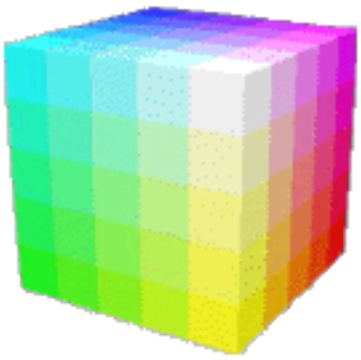


Definition: The **Map** action maps the value of an agent's **Attribute**, a **Simulation Property**, or a Visual AgenTalk **Formula** to a **Color** that lies between two user-specified colors along the color line. The range of the attribute's values to be mapped to a color is specified with numbers or **Visual AgenTalk Formulae**.

To understand how this action works, you must first understand a little about **color cubes** and about how humans see color. In general, humans see colors in a three-dimensional color space. Humans have receptors that see red, green, and blue pigments.

You can create any color visible to a human by mixing different spectrums of red, green, and blue pigments together to form a color. Thus, a **color cube** has x, y, and z axes which represent amounts of red, green, and blue pigment.

Points in a color cube represent colors containing different amounts of red, green, and blue pigment. Black is the color we see when there is no red, no green, and no blue pigment present. White is the color we see when the maximum amount of pigment our eyes can detect is present for red, green, and blue pigments. An image of a color cube is shown below.



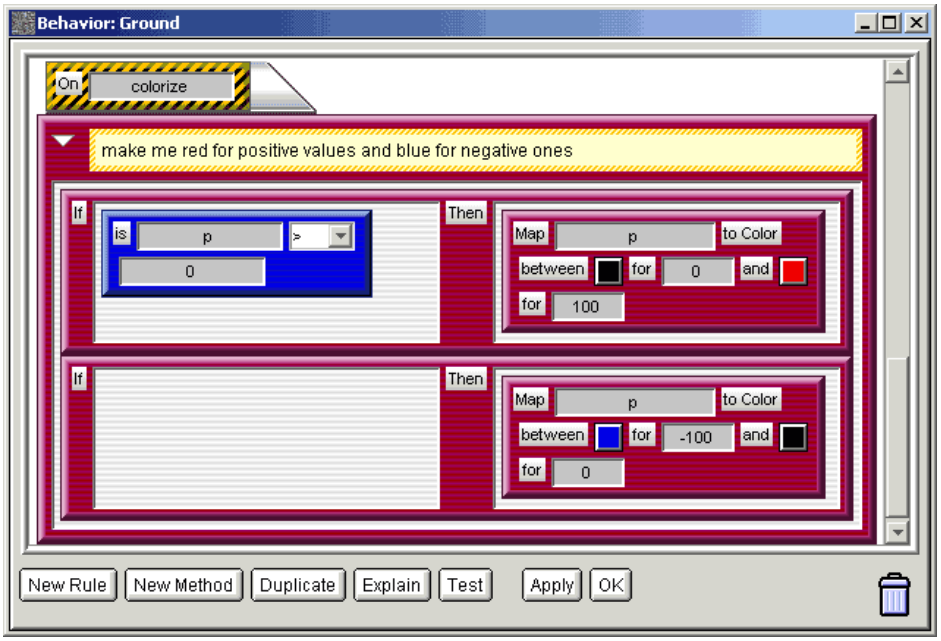
A user specifies points in a color cube using the **Color** parameter. The two points/colors become endpoints for a color line that stretches between them in the color cube.

A user also specifies minimum and maximum numerical values for the attribute to be mapped using numbers or **Visual AgenTalk Formulae**. These values are mapped to the color endpoints previously mentioned.

As the values of the chosen attribute pass through the user-specified numerical range, the **Map** action colorizes the agent with a color that corresponds to a point along the color line between the two color endpoints.

Parameters: **attribute**, **color**, **formula**

Example: The Ground agent in the **Ultimate Pacman** simulation colorizes the diffusion values. Negative values get mapped to a shade of blue, positive values get mapped to a shade of red.



See the Ultimate Pacman simulation on the AgentSheets CD-ROM.

Plot Attribute Action

Definition: The **Plot Attribute** action plots the value of the **Attribute** named in the action's text field between endpoints as specified with numbers or **Visual AgenTalk Formulae**.

The attribute value is plotted using a color displayed in the action's left color field on a background color specified in the action's right color field.

Attribute values are plotted along the y-axis of a 2-D plot window.

Time is represented along the plot window's x-axis. As time passes, plots are marked to the right of the rightmost pixel marked during the previous time step.

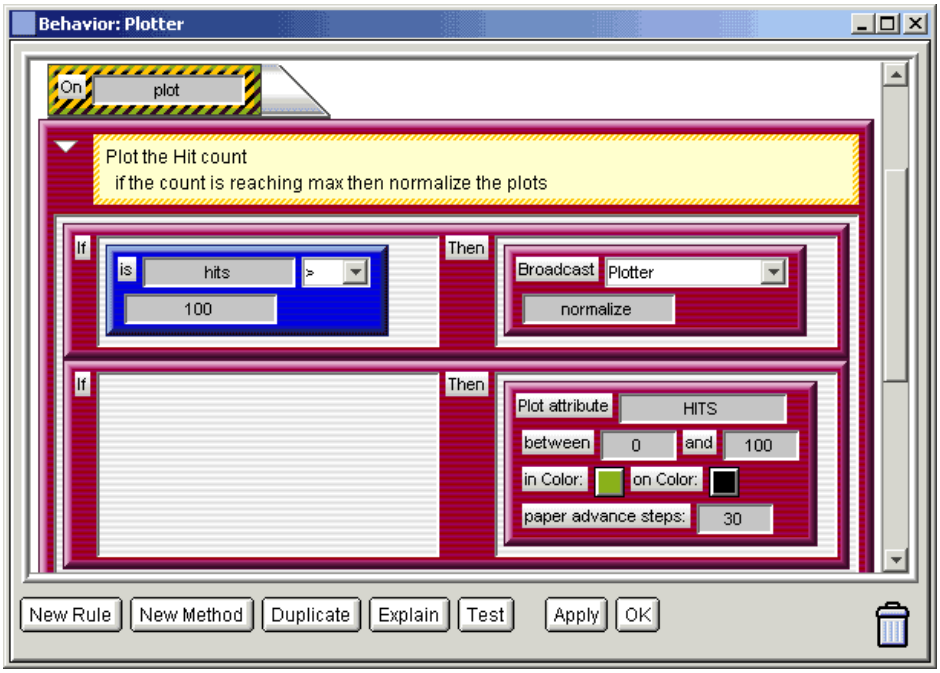
This can be understood metaphorically as "advancing the plot paper" a certain distance at each time step. The number of steps the plot advances the "paper" is specified by a number or a Visual AgenTalk formula.

As the plotter drags across the plot paper rolling underneath it, time is represented as lines drawn from left to right (x-axis) while plotted values are read in a vertical direction (y-axis).

The plot window where plots are drawn replaces the agent's depiction. A plot window is as big as the agent that executes the **Plot Attribute** action.

Parameters: **attribute**, **formula**

Example: The Plotter agent in the **Counting Pachinko** simulation plots the number of balls hitting it between 0 and 100, using a green color on a black background. Each time it is hit, it advances the plotting "paper" 30 steps to simulate the effect of a bar going up every time it plots.



See the Counting Pachinko simulation on the AgentSheets CD-ROM.

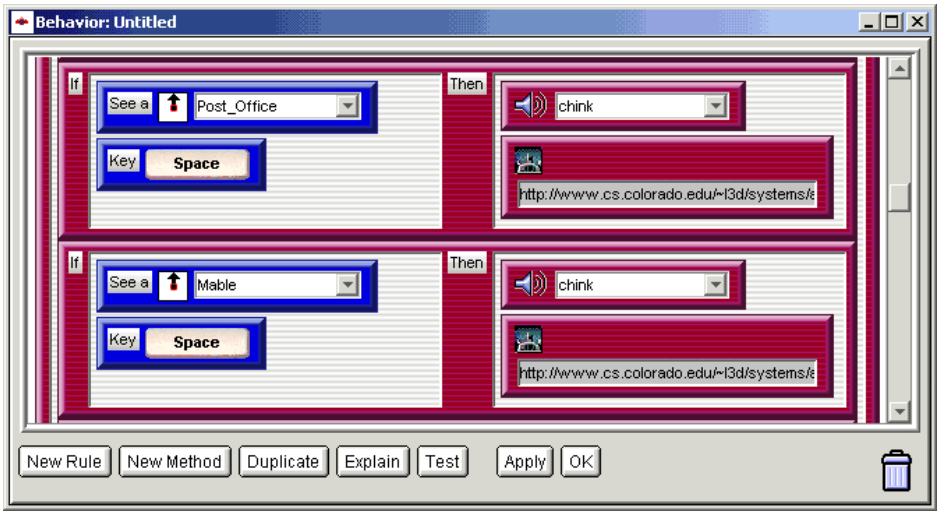
Open URL Action

Definition: The **Open URL** action opens the **URL**. The **URL** is specified in the text field and can identify a web address, a file address, an FTP address or an e-mail address. If a web, file, or FTP address is specified, it will be opened in a browser. If an e-mail address is specified, it will be opened in your default mail program.

If no browser is running, the system will open the default Web browser.

Parameters: **URL**

Example: In the **Grotesque City**, the character opens the appropriate web pages when he encounters buildings that have stories associated with them (the space bar needs to be pressed).



See the Grotesque City simulation on the AgentSheets CD-ROM.

Load Background Action

Definition: The **Load Background** action loads a file specified by the **image file name** parameter into a Worksheet as a background graphic.

AgentSheets can recognize GIF, JPEG, and PNG file formats.

Backgrounds are required to be located in the "backgrounds" folder inside the "applet" folder of a project (Please refer to **Anatomy of a Project** for details on the structure of the project folder). If you load a background that does not already exist in the backgrounds folder, the system will copy it there for you.

Parameters: **image file name**

Ristretto Information: The background file must be provided in GIF or JPEG format so that it can be loaded in the applet.

Set Direction Action

Definition: The **Set Direction** action defines the direction in which the agent is heading. The direction is determined by the **Direction** parameter.

Note: This action does not actually move the agent.

Parameters: **direction**

Stop Simulation Action

Definition: The **Stop Simulation** action stops the simulation if it is running. This has the same functionality as pressing the **Stop** button on the worksheet.

Parameters: **None**

Reset Simulation Action

Definition: The **Reset Simulation** action resets the simulation, that is it resets the worksheet to its last saved version. This has the same functionality as pressing the **Reset** button on the worksheet.

Parameters: **None**

Clear Simulation Action

Definition: The **Clear Simulation** action clears the worksheet that is currently running. This has the same functionality as pressing the **Clear** button on the worksheet.

Parameters: **None**

Triggers

Definition: The **Triggers Palette** contains the following **Triggers** that label method boxes in the Visual AgenTalk language behavior editors:

While Running
On
Tool
When Creating A New Agent

One of these triggers is at the head of every method and determines when the method will be called. For example, any agent's **While Running** method will be called at each time step **while** the agent is **running** its behavior program.

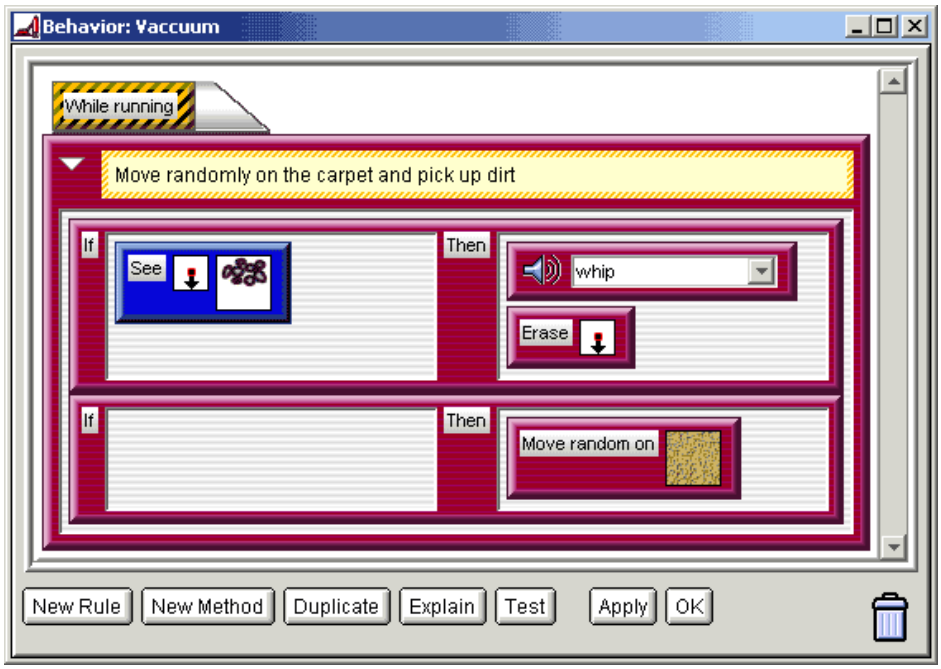
A behavior editor can contain any number of methods. A method is the combination of a trigger and a set of rules. A new behavior editor comes up with a single method containing the While Running trigger and one empty rule.

While Running Trigger

Definition: The **While Running Trigger** trigger is called once every simulation cycle while the simulation is running.

Parameters: none

Example: While the **Vacuum Cleaner** simulation is running, the Vacuum agent picks up dirt if it sees it in the cell below, otherwise it just moves randomly on the floor tile.



See the Vacuum Cleaner simulation on the AgentSheets CD-ROM.

On Trigger

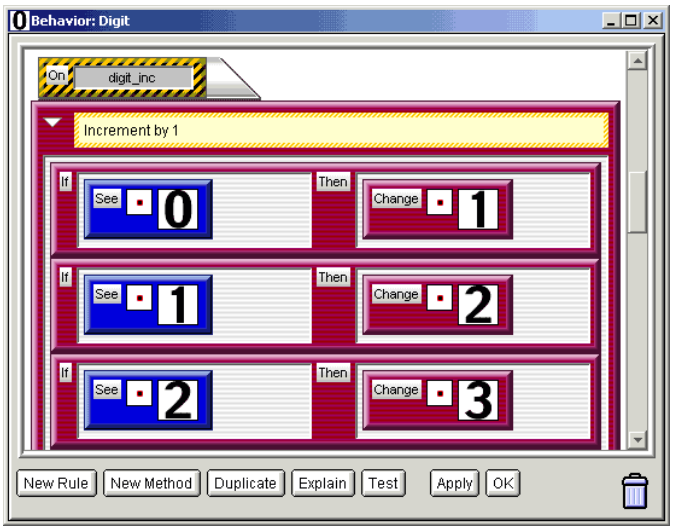
Definition: **On** triggers allow you to name methods that will later be called. The method is called when the appropriate message is sent to the agent via a **Make** action or a **Broadcast** action from itself or from another agent. The message sent needs to be a **valid method name**.

Upon receiving a message, an agent calls the method referred to in the message and carries out the actions associated with the first rule whose conditions are all met.

When programming an agent, it is recommended that you divide the agent's behavior into "chunks." It is natural to do this when describing an agent's activity in words. At different times an agent might walk, run, sit, speak, or act in any number of ways. If you focus on walking, you may wish to create one method to describe how an agent walks. You may even wish to create another method to describe how an agent takes a step. We suggest that you divide groups of rules into named methods that make sense for the agents in your specific simulation.

Parameters: **Method Name**

Example: In the **Pascal's Adding Machine** simulation, the **On** trigger is called when the Digit agent gets the "Digit-Inc" message, which causes the number to increment by one.



See the Pascal's Adding Machine simulation on the AgentSheets CD-ROM.

Tool Trigger

Definition: The **Tool** trigger is used to program an agent to respond to clicks with the worksheet tools.

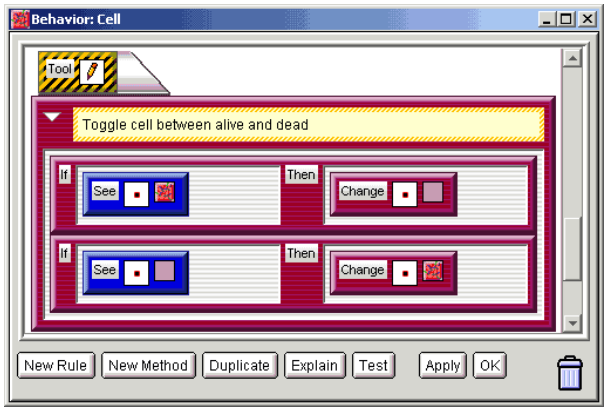
The rules located in a method labeled with a **Tool** trigger are executed whenever the executing agent senses a click from the designated tool, whether the simulation is running or not.

Warning: If you redefine the **Arrow**, **Eraser**, and **Pencil** tools, you will overwrite system functionality as follows:

- **Arrow** - You will be unable to select the agent in the worksheet, if that agent that redefines the Arrow tool.
- **Eraser** - You will be unable to erase an agent in the worksheet, if that agent redefines the Eraser tool.
- **Pencil** - Clicking on the agent that redefines the Pencil tool will execute the rules you specified under the Tool trigger and will not add the agent selected in the gallery to the worksheet. Clicking anywhere else with the Pencil tool will still allow you to add the agent selected in the gallery to the worksheet.

Parameters: **Tool**

Example: When the **Pencil** tool is used on the **Cell** agent in the **Game of Life** simulation, it toggles between the dead and alive depictions.



See the Game of Life simulation on the AgentSheets CD-ROM.

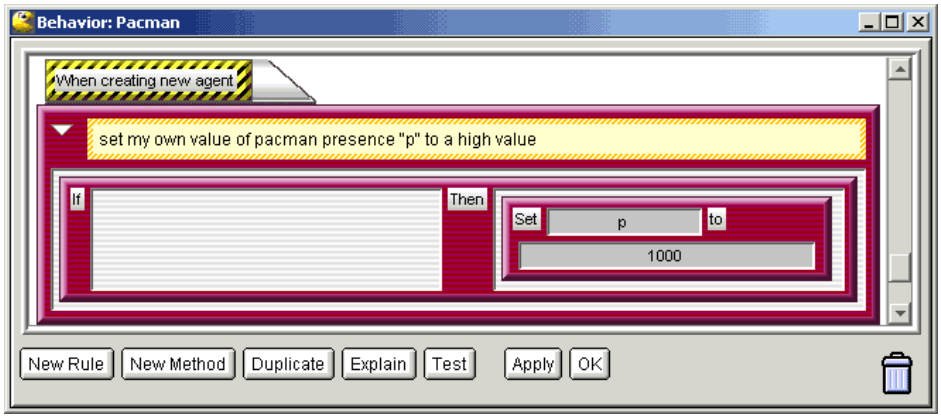
When Creating New Agent Trigger

Definition: The **When Creating New Agent** trigger executes the enclosed rules whenever a new agent is created in a worksheet. These rules are executed regardless of whether the worksheet in which the new agent is created is running or not.

The **When Creating New Agent** trigger is usually used to initialize the agent's state.

Parameters: none

Example: In the **Ultimate Pacman** simulation, when a new Pacman agent is created, it initializes its *p* value to 1000.



See the Ultimate Pacman simulation on the AgentSheets CD-ROM.

Parameters

Definition: A **Parameter** is an item of information such as a name, a number, a sound, a color, or a selected depiction which affects the operation of a condition, action or trigger command in Visual AgenTalk. For instance, the **Sound** parameter in the **Play Sound** action allows you to select a specific sound to be played.

When running AgentSheets use the **Help** | **Show Balloons** menu option or the **Explain** button to explore parameters.

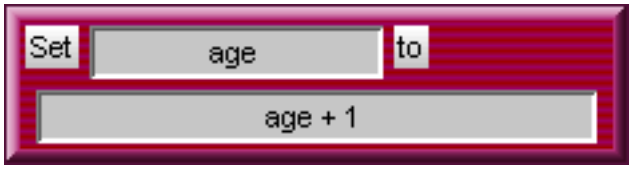
The following parameters are available in AgentSheets:

Attribute
Class
Color
Comparator
Depiction
Direction
Formula
Image File Name
Key
Method Name
Sound
String
Tool
URL

Attribute Parameter

Definition: The **Attribute** parameter identifies an editable name, e.g., "energy", which refers to an agent characteristic. The attribute name provided needs to be a **valid name**.

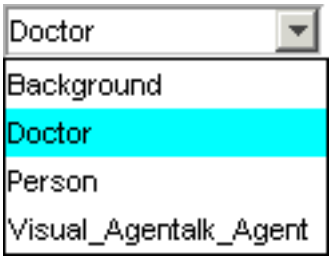
The **Attribute** parameter (called **age**) is shown below as part of the **Set** action.



Class Parameter

Definition: The **Class** parameter contains a list of all the agent classes housed in the current Gallery. This corresponds to a list of all **Agents** in a Gallery.

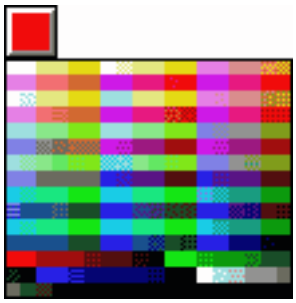
An example of an extended **Class** menu is shown below.



Color Parameter

Definition: The **Color** parameter allows users to select different colors visually.

Choose any color by holding clicking on the color parameter and selecting a color from the 256 colors in the **Color Palette** that pops up when a color parameter is extended:

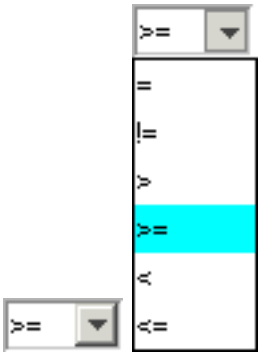


The new color is set when you release the mouse button and is shown in the collapsed color parameter.

Comparator Parameter

Definition: The **Comparator** parameter contains a list of different numerical comparison operators. These operators are often used to compare attribute values within and between agents.

A collapsed **Comparator** parameter is shown on the left and an extended **Comparator** pop up menu is shown on the right.



Depiction Parameter

Definition: The **Depiction** parameter identifies an agent by the way it looks.

When you click on a **Depiction** parameter, a window containing all the depictions located in the current **Gallery** pops up.

Choose any depiction by holding down the mouse button and highlighting it. Let go of the mouse button and the highlighted depiction will be displayed in the depiction window of the command.

A collapsed (left) and extended (right) **Depiction** pop-up menu is shown below:

Direction Parameter

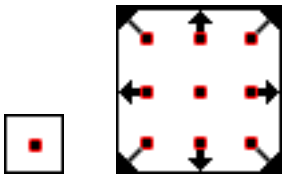
Definition: The **Direction** parameter specifies a direction. The **Direction** parameter values correspond to the Moore/8-cell neighborhood model. The dot (.) refers to the agent itself.

Click on the operator in its collapsed state to extend it. Then choose a direction by highlighting the preferred direction.

If the **Direction** parameter is used in a **Condition,** it will specify which neighboring cell of an agent (or itself) should be tested when executing a condition check.

If the **Direction** parameter is used in an **Action**, it will specify which neighboring cell of an agent (or itself) the action will be performed on.

A collapsed (left) and extended (right) **Direction** pop-up menu is shown below:



Formula Parameter

Definition: The **Formula** parameter allows users to create spreadsheet-like formulae.

Within the formula field you can call any mathematical functions supported by **AgentSheets** and **Ristretto**. This allows arbitrarily complex relationships to be set up using **Mathematical Operators** that define relationships between attribute or simulation property values within and between interacting agents.

An example of a **Formula** parameter is shown below in the context of the **Set** action, where the pollution attribute is being set to the average of the pollution of the agents around it: pollution = (pollution[left] + pollution[right] + pollution[up] + pollution[down]) / 4

Formula Syntax

Definition: Visual AgenTalk (VAT) Formula Syntax is used to create VAT formulas. The following Formula Syntax is accepted by **Ristretto** :

	Operation	Definition	Example	Result
Arithmetic Operations	x + y x - y x * y x / y	Basic arithmetic operations	3 + 4	7
Agent Attribute Access	*attribute*	Access the value of an agent **attribute**. An agent can have any number of attributes defined by the user	Diameter Diameter * 3.14	value of the agent attribute "Diameter" value of attribute "Diameter" multiplied by 3.14
Remote Agent Attribute Access	*attribute*[up] *attribute*[down] *attribute*[left] *attribute*[right] *attribute*[top] *attribute*[bottom] *attribute*[row, col]	Access the value of other agents' attribute using relative coordinates. Valid coordinates are up, down, left, right, top and bottom. Coordinates can also be specified numerically as row, column. Valid values for row and column are -1, 0, 1. Positive row values indicates right, positive column indicates down.	age[left] age[top] Temperature[-1,-1] 0.25 * (Temp[left] + Temp[right]+ Temp[up] + Temp[down])	value of the "Age" attribute of agent to the left value of the "Age" attribute of the agent on top value of the "Temperature" attribute of the agent above to the left The average of the value of the "Temp" attribute of the agents left, right, up and down.
Simulation Property Access	*@simproperty*	Access the value of a global simulation property **Simulation Properties** are used to share information between agents. Users can inspect and edit the values of simulation properties using the simulation property editor. The "@" sign is used to differentiate the simulation properties from agent attributes.	@Time	value of simulation property "Time"
Trigonometric Functions	sin(x)	Trigonometric function sine, where x is expressed in radians	sin(3)	0.1411
	cos(x)	Trigonometric function cosine, where x is expressed in radians	cos(3)	-0.9900
	tan(x)	Trigonometric function tangent, where x is expressed in radians	tan(3)	-0.1425
Random Number Generator	random(number)	Returns a pseudo random number between zero and number . AgentSheets differentiates between integers and decimal numbers. If the number is an integer (e.g. 4), an integer between 0 and number is returned, whereas if number is a decimal number, a decimal number between 0 and number is returned.	random(4.0) random(4)	returns decimal number between 0 (inclusive) and 4.0 (exclusive) returns either 0.0, 1.0, 2.0, or 3.0

Other	x ^^ y	Exponentiation function. Raises the base number x to the exponent y	15^^4	50625
	sqrt(x)	Square root function. Takes the square root of number x, where x is a non-negative integer or a decimal number (x>=0).	sqrt(256)	16
	x % y	Modulo function. Gives the remainder of the division of x / y.	17 % 4	1

Image File Name Parameter

Definition: The **Image File Name** parameter specifies an image file name.

An example of the **Image File Name** parameter is shown below, used to load a background image into a **Worksheet** .

Key Parameter

Definition: The **Key** parameter indicates a letter, number symbol, or function key on the keyboard.

To define the key that should be tested, click on the button next to the **Key** label and press the desirable key on the keyboard.

An example of the **Key** parameter is shown below in the Key condition where the "space" key must be pressed to satisfy the test.

Method Name Parameter

Definition: The **Method Name** parameter is a string used to name a group of rules in an agent behavior.

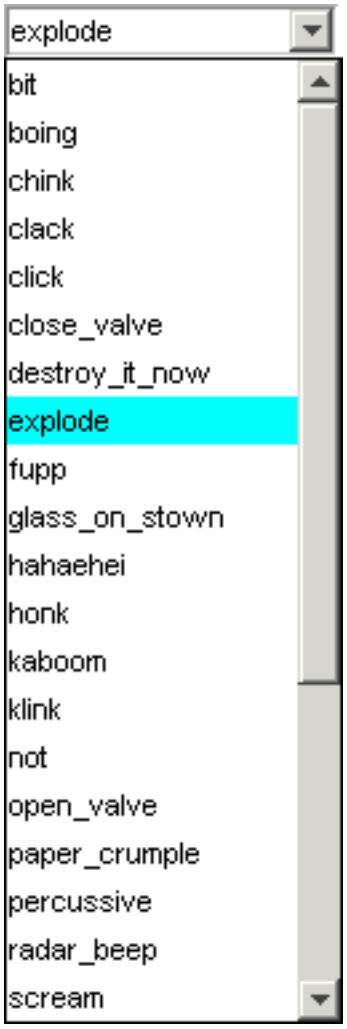
The string must be a **valid method name**. The method name may be used to send a message to an agent via the **Make** or the **Broadcast** action to trigger the corresponding **On method**.

A **Method Name** parameter ("animate") is shown below used in the **On trigger**.

Sound Parameter

Definition: The **Sound** parameter is a list of prerecorded sound effects available to **AgentSheets** .

An extended **Sound** parameter menu is shown below:



String Parameter

Definition: The **String** parameter identifies a word or phrase.

When the string parameter is used in the **Say** action, it can contain references to agent **Attributes** or **Simulation Properties**. For example, if an agent has an attribute called "age" whose value is currently 11, and the String in the **Say** action is "I am ~age years old", then the Say command would print "I am 11 years old" in the worksheet's message area.

When the string parameter is used in the **WWW Read** condition, it is a word or phrase that is used to search the specified web page. The string, in this case, can contain wildcards ("*").

An example of the String parameter ("hello") is shown below as it is used in the **Say** action:

Tool Parameter

Definition: The **Tool** parameter identifies a **Worksheet Tool**.

A collapsed (left) and extended (right) **Tool** parameter is shown below:

URL Parameter

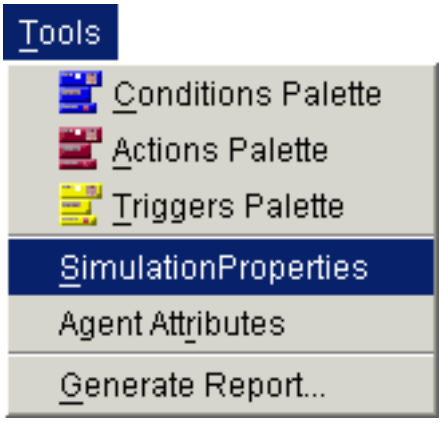
Definition: The **URL** parameter identifies a Universal Resource Locator, which is a pointer or address that corresponds with a World Wide Web resource.

The **URL** may identify a Web address, a file address, an FTP address, or an e-mail address. For example:

http://<web page address>
ftp://<ftp server address>
file:///<local file address>
mailto:<email address>

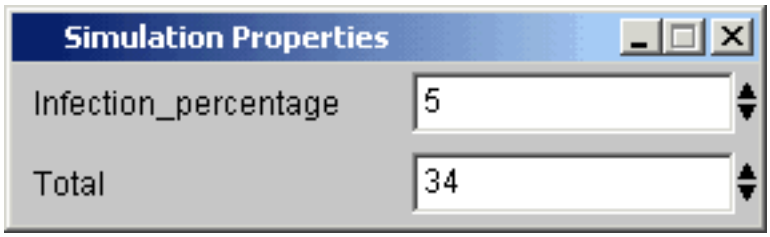
An example of the *URL* parameter is shown below in the **Open URL** action.

Simulation Properties



Description: **Simulation Properties** are used to share information between agents. Unlike **Agent Attributes**, which are local to each instance of an agent, **Simulation Properties** are global values accessible by all agents. They can be numbers used as inputs or outputs for a simulation.

Users can inspect and edit the values of simulation properties using the **Simulation Property Editor**, shown below.

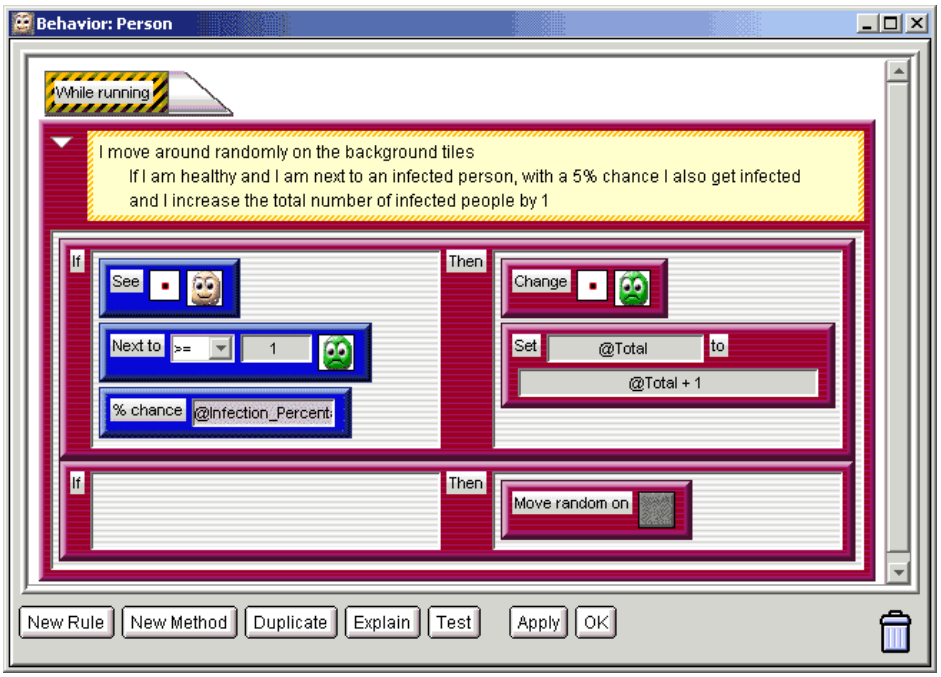


You can edit the value of a **Simulation Property** by editing its value field. You can also use the Up and Down arrows to increase or decrease the value of a specific **Simulation Property**.

You do not have to explicitly define a new Simulation Property in the editor. You can just use a simulation property in an agent's behavior. Even though you can use any name for your simulation property, it is advised to use a **valid name** for compatibility with the Macintosh version.

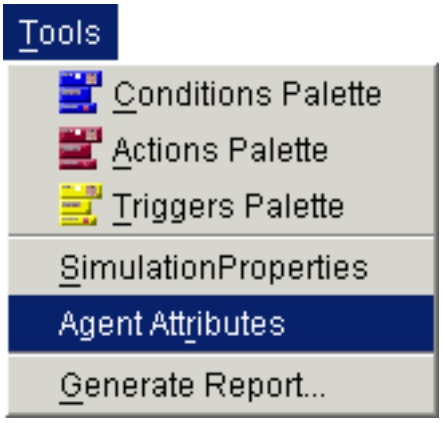
Agents can read and write Simulation Properties. For example, in the **Virus Attack** project, two Simulation Properties are defined, as shown in the behavior editor below:

- **Total**: a counter for keeping track of the total number of sick people. Every time a Person agent gets sick, it writes the "Total" Simulation Property using the **Set** action, incrementing it by 1, as shown in the first rule below.
- **Infection_Percentage**: the percentage a healthy Person has of acquiring the virus. **%Chance** reads the **Simulation Property** and uses its value, as shown in the first rule below:



The "@" sign is used to differentiate **Simulation Properties** from **Agent Attributes**.

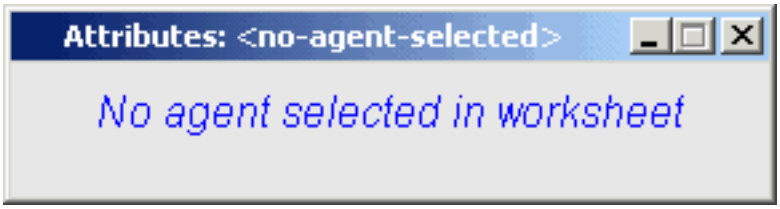
Agent Attributes



Description: **Agent Attributes** are used to specify information about an agent. Unlike **Simulation Properties**, which are global to the entire simulation, Agent Attributes are local to each instance of an agent.

Users can inspect and edit the values of agent attributes properties by selecting an agent in the worksheet and choosing Tools | Agent Attributes. If the selected agent has attributes, they will show up in the **Attributes Editor**. The one shown below contains an agent attribute called "age" whose value is 27.

If the selected agent does not have attributes, the following message will appear in the Attributes Editor.



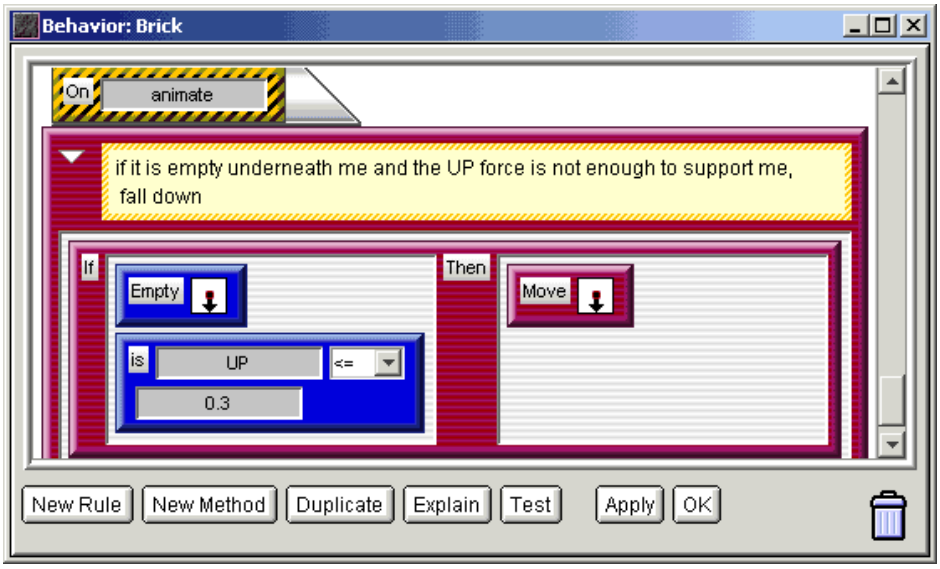
You can edit the value of an **Agent Attribute** by editing its value field. You can also use the Up and Down arrows to increase or decrease the value of a specific **Agent Attribute**.

You do not have to explicitly define a new Agent Attribute in the editor. You can set it using the **Set** action in an agent's behavior.

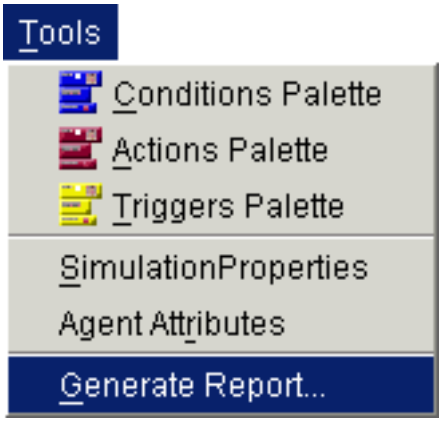
Even though you can use any name for your agent attribute, it is advised to use a **valid name** for compatibility with the Macintosh version.

For example, in the **Bridge Builder** project, the Brick drops down if the cell below it is empty, and the force acting on it is less than or equal to 0.3 (and therefore

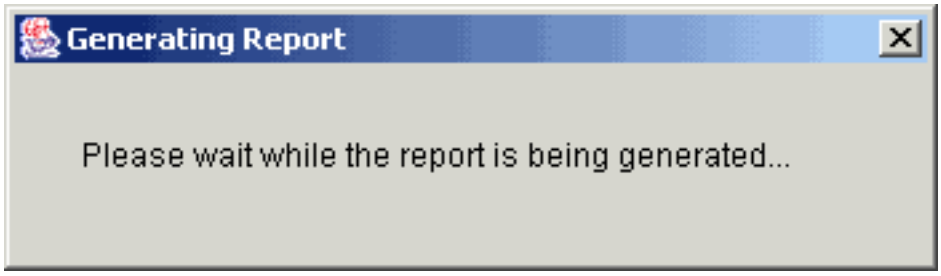
cannot hold it in place). The force is represented as an agent attribute called "UP", as shown in the behavior editor below.



Generate Report

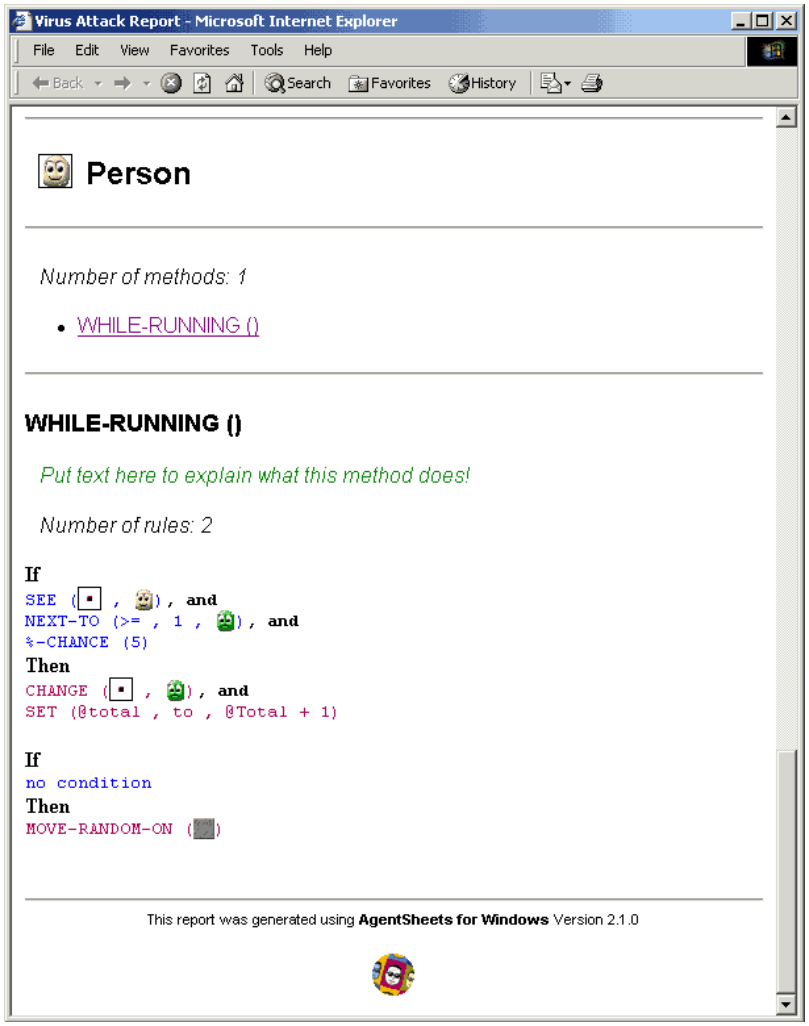


Description: The **Generate Report...** menu option is used to create a web page containing a comprehensive report about the current project containing agent descriptions and behavior descriptions for all the agents in a project. When the Generate Report menu option is selected, the following dialog appears, prompting you to wait for the report generation to finish.



Once the generation is complete, the generated report will be opened in a browser as shown below.

The generated report also includes behavior descriptions for each of the agents as shown below for the agents of the Virus Attack project.


Virus Attack Report - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Person
Number of methods: 1
WHILE-RUNNING ()
WHILE-RUNNING ()
Put text here to explain what this method does!
Number of rules: 2
If
SEE (,), and
NEXT-TO (>= , 1 ,), and
%-CHANCE (5)
Then
CHANGE (,), and
SET (@total , to , @Total + 1)
If
no condition
Then
MOVE-RANDOM-ON ( )
This report was generated using AgentSheets for Windows Version 2.1.0