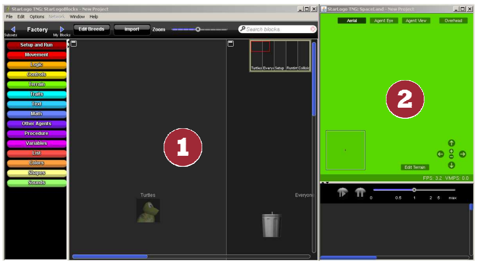
StarLogo TNG Quick Start Guide

Starlogo The Next Generation (TNG) is an agent-based, graphical programming environment designed for teachers and students to study and create 3-D games, models and simulations.

Installing and starting the software

PC:

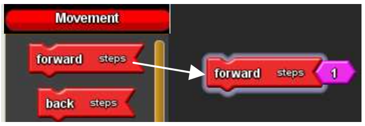
Mac:



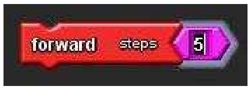
Window Name	Contains	Function
1 StarLogoBlocks	Palette of blocks Canvas	**Blocks** = programming commands **Canvas** = place to drag and drop blocks to build programs
2 Spaceland	Spaceland Runtime Box	**Spaceland** = 3-D virtual world that consists of a plane (terrain) and agents (characters, objects) that execute the blocks. **Runtime Box** = controls the speed of the running programs; programmable user-interface buttons

First Program

1. In the Spaceland window, click on the ⊕ icon to zoom in until you can see the two turtle agents clearly.
2. Click on Movement drawer tab.
3. Position mouse cursor over the forward block. Hold down left mouse button and drag the block to the canvas. Release the mouse button to drop the block.



4. Double click on the forward block and watch the turtle agents move forward 1 step. Note that both of the agents execute the same instruction.
5. If desired, repeat step 4 a few more times.
6. Click on the pink 1 block to highlight it and change the number to a higher number, like 5.



7. Double click the forward block to see the turtle agents move forward the number of steps that you specify. Click the ⊖ icon to zoom out if needed.

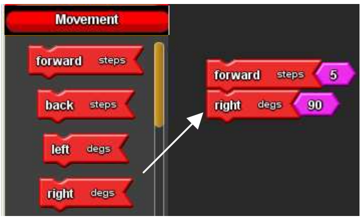
You have learned:

- **to drag blocks from the palette and drop them in the canvas**
- **one way of telling the agents to execute a block instruction**
- **to change the value of a pink number block**
- **one way of zooming in and out of Spaceland**

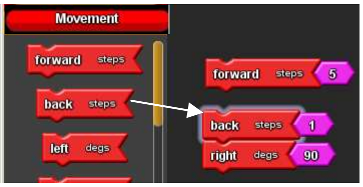
Go to File menu > Open projects > StarLogo TNG\projects folder to see examples of games and simulations. For most people, the path is: C:\Program Files\StarLogo TNG\projects

Program a Stack of Blocks

1. From the Movement drawer, drag out a right block and position it just below the forward block. When you drop the block, the forward block and the right block should snap together to make a single stack.
2. Double click on the stack and watch the turtle agents execute the movement.
3. Repeat step 2 several times to see the agents move in a square pattern.



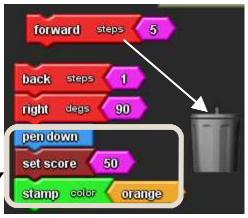
4. Replace the forward block with back block:
   - i. Click and drag the right block to separate it from the forward block.
   - ii. Go to the movement drawer to pull out a back block and drop it near the top of the right block so that it attaches to the right block.
   - iii. Drag and drop the forward block in the trash can in the bottom right corner of the canvas.



You have learned how to:

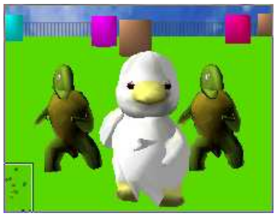
- **connect blocks together to make a stack**
- **get the agents to execute a stack of blocks**
- **replace a block in a stack**
- **delete a block**

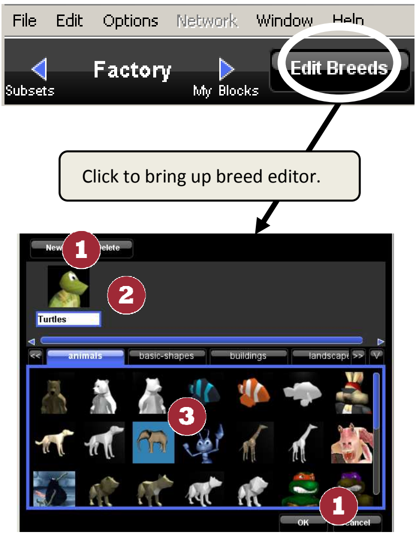
Feel free to explore the drawers and try connecting different blocks together.



Breed Editor

SL-TNG is agent-based programming. The default breed is named Turtle and the agent looks like a 3-D rendering of a turtle. You can have many agents that belong to the Turtle breed. When programming, you would typically give instructions to an entire breed, not individual agents. Thus, normally, all Turtle agents would execute the same instructions at the same time (so it appears). Below is a screenshot showing several breeds of agents.



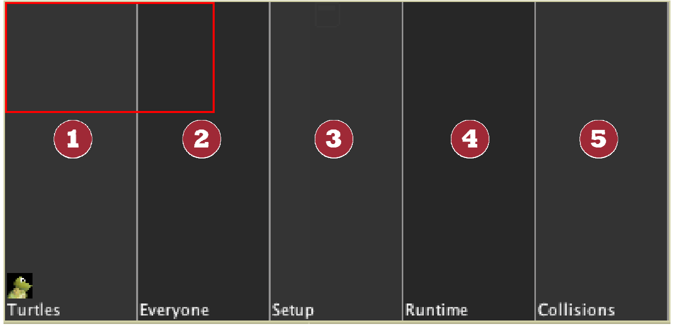


Click to bring up breed editor.

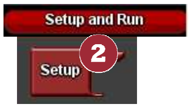
1. **New** = create new breed; **Delete** = delete breed; **OK** = save changes and close editor; **Cancel** = don't save changes and close editor
2. Shows all existing breeds. Click on a breed to edit its shape and name.
3. Palette of shapes available, arranged by category. Use scroll bar to show more shapes.

Try creating a new breed and pick a new shape and name for it.

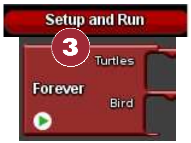
Canvas Sections



1 Each breed has its own section of the canvas. You can use this page to define variables and procedures for particular breeds of agents.

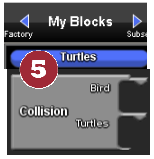


2 Declare variables and procedures that are shared by some or all breeds here.



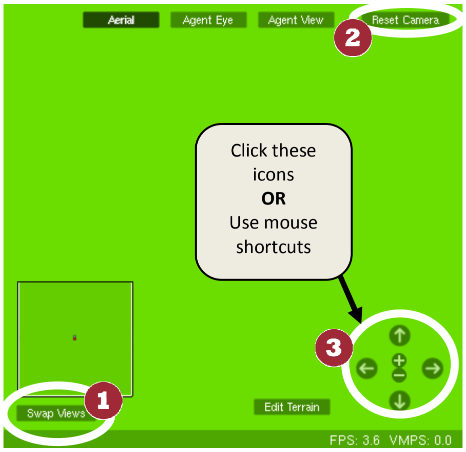
3 Put the Setup block(s) here. Setup is used to set the starting conditions, such as creating and placing agents, setting the initial values of variables, etc.

4 Put the Forever or Run block(s) here. Each breed has its own section in the Forever block. Agents execute instructions in the Forever block continuously.



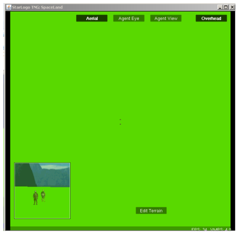
5 Put Collision blocks here to program how agents behave when different breeds “collide” or touch each other in Spaceland.

Spaceland Camera



Click these icons **OR** Use mouse shortcuts

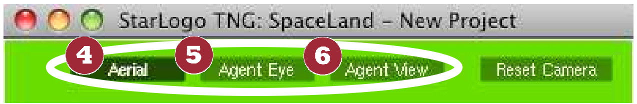
1 **Swap Views**
-click to switch between 2D and 3D views
-2D shows ALL of Spaceland
-camera cannot move in 2D



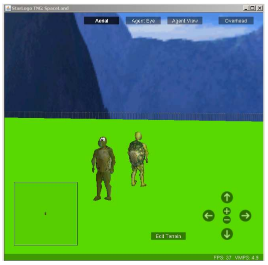
2 **Reset Camera**: Click to reposition camera to default

3

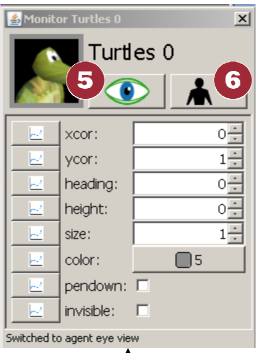
Function	Click Icons	Mouse shortcuts
Zoom in and out	⊕ ⊖	Rotate scroll wheel (up=zoom in; down=zoom out)
Pan	← → ↑ ↓	Hold down left button and move the mouse
Rotate	Not available	Hold down right button and move the mouse



4 **Aerial** (default) = 3-D
-can move camera



Agent Monitor = displays trait values of a particular agent



Monitor Turtles 0

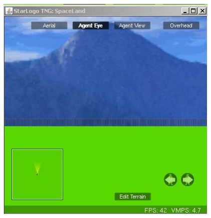
Turtles 0

xcor:	0
ycor:	1
heading:	0
height:	0
size:	1
color:	5
pendown:	□
invisible:	□

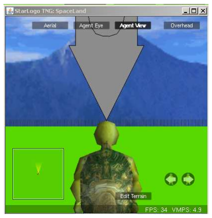
Switched to agent eye view

Click on an agent.

5 **Agent Eye** = 1st person point of view

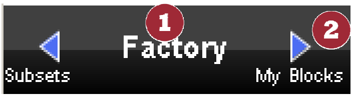


6 **Agent View** = over the shoulder



Try changing the values of the traits in the agent monitor. Watch what happens to the agent.

About Selected Palette Drawers



1 Factory Palette

Setup and Run	**Movement**	**Terrain**
Blocks that set the initial conditions of a program and run the program	Blocks for moving the agents	Blocks for changing the terrain
Traits	**Logic**	**Variables**
Pre-defined variable getters and setters for agentssuch as color, shape, breed, etc.; values can be seen in the agent monitor	If/then, boolean, and other flow control structures	Blocks to define variables
Math	**Controls**	**Procedure**
Blocks to construct mathematical expressions	Blocks that control input from user	Blocks to define procedures

2 My Blocks Palette

- This palette contains dynamically generated drawers and blocks as new breeds, variables, and procedures are defined
- Examples of blocks include: collision blocks, variables getters and setters, and procedure call blocks

Check out more tutorials at http://www.imaginationtoolbox.com