

Treasure Hunt Game Programming Activity

An extensible, build-from-scratch, game programming activity in StarLogo TNG

This activity is designed for people with some experience working with and extending StarLogo TNG projects who would like to learn how to build a new project from start to finish. Rather than present step-by-step instructions, it contains a series of small challenges paired with some commands that are useful for completing the challenges. The result of all of the work to complete the challenges is a fully functional, extensible treasure hunt game.

Goals

- Create a treasure hunt game in StarLogo TNG
- Learn to start a project from scratch
- Explore new commands without fear of breaking something
- Learn essential concepts that apply to every SLTNG game

Instructions

Read each challenge and examine any commands that are given to help complete the challenge. Then try to complete the challenge any way you want to. Don't be afraid to explore and make mistakes! Use any commands or features you want to use, but if you're stuck make sure to look again at what is supplied with the challenge. Feel free to ask your neighbors for help, but always feel free to make one mistake first.

Challenges 1-5 will lead to a basic, but fully functional game. Challenges 6-10 extend the basic game and teach the use of more blocks. There are programming tips scattered throughout the activity sheets. If you don't have time to complete the remaining challenges, you may still want to read the programming tips, which are relevant when you are programming any StarLogo TNG project, not just for this activity.

Game Description

The player controls an agent using keyboard arrow keys to collect as many treasure agents as possible in 90 seconds, while avoiding obstacles.

IMPORTANT NOTE: There are blocks provided with every challenge. These blocks are only suggestions to complete the challenge. IT IS NOT NECESSARY TO USE ALL THE BLOCKS PROVIDED.

Challenge 1: Choose and Create your game objects/player

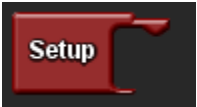









Create a player breed, treasures breed, and obstacles breed. You may replace the default turtles breed or delete it. Select shapes for each and name them.

Program Setup to create 1 player agent, many treasures, and some obstacles. Scatter some or all of your agents.

Useful features/commands:



: Opens the Breed Editor.

Block	How it works	Drawer
	Usually goes in the Setup section of the canvas. <u>Used to set up initial conditions of the program</u> , such as creating agents, scattering agents, clearing agents from previous run, etc. Creates a button in the runtime window that you can click to execute the setup instructions. You can have more than one setup block. Each setup block has its own button in the runtime area of the Spaceland window.	
	Used in the Setup block to delete all agents only – doesn't affect terrain – from previous run.	
	Used in the Setup block to create new agents of a particular breed. The pink argument block specifies how many agents should be created. Default position of the created agents is in the center of Spaceland. The create block with the do section allows for other instructions for the batch of created agents to do right away – such as being placed in a certain location, have a certain color or size, etc.	
	May be used in the Setup block to randomly place agents of a particular breed that have been created in spaceland, with height zero.	
	May be used in the Setup block to randomly place agents of all breeds that have been created in Spaceland, with height zero.	











Test your Setup block by clicking on the Setup button in the Spaceland window to see if Setup is working as you intended.

Programming Tip
Make small changes and test

Example: Add a block to the setup block and then click the setup button to see what happens.

Challenge 2: Set camera to 1st person perspective

To start the game in 1st person perspective of the player agent, set the camera to follow the player agent, and set the camera angle to over the shoulder or agent eye.

Block	How it works	Drawer
	Contains the ID number of the agent that executes the block.	
	Sets the camera to show the perspective of a particular agent. Takes a number argument that represents the ID of the agent.	
	Sets the view angle of the camera belonging to the most recent selected agent to over shoulder (or agent view).	
	Sets the view angle of the camera belonging to the most recent selected agent to agent eye.	
	Sets the view angle of the camera to aerial, the 3-D view from some point in the air.	

Sample Code:



Test your Setup block by clicking on the Setup button in the Spaceland window. You should be looking over the shoulder of your player agent.

Programming Tip

Use drop down menu on some blocks to see related blocks

Example: If you mouse over the over shoulder block, a small triangle will appear on the right edge. Click on this to display the drop down menu to see related blocks, such as agent eye.

Switch Driver/Navigator roles!

Challenge 3: Keyboard Controls

Recall that procedures are useful when you want to use a set of instructions multiple times or when you want to break down a complicated task into discrete parts and program each part separately in an easy-to-read way.

There are two important parts to using procedures:

- Defining a procedure: adding instructions to the procedure block. It is good programming practice to name the procedure for personal organizational purposes.
- Calling a procedure: using the procedure after it has been defined (for example, in a run or forever loop)

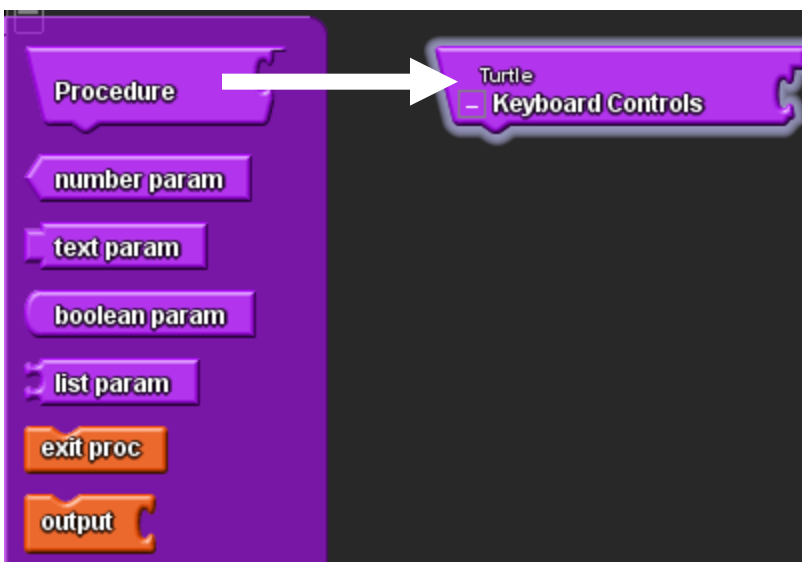
Programming Tip

Rename blocks to make the programming code easier to understand or to provide meaningful information to the user.

All blocks that generate buttons can be renamed, such as setup, forever, monitor, and line graph. Procedure definition and Variable declaration blocks can also be renamed.

Example: Click on the run text in the run block to select it and replace with something like “play game” (no quotes).

Define the procedure in StarLogo TNG by dragging a Procedure block from the Procedure drawer to a page on the canvas. Click on the Procedure name and change it to Keyboard Controls.

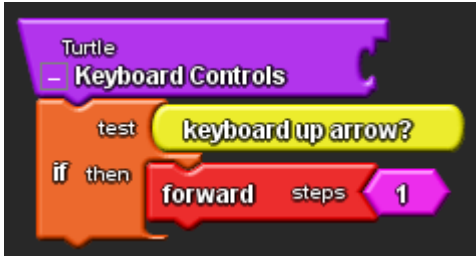


Programming Tip

Which page should I use to define procedures?

Define procedures on the page of the breed that will follow the procedure. If multiple breeds follow the procedure, define it on the Everyone page.

Under the Procedure block, attach the following blocks:



These blocks read: Agent tests to see if the up arrow key is pressed. If true, go forward 1 step.

Program additional blocks to do the following:

- Down arrow = Back 1
- Left arrow = Turn left 5 degrees
- Right arrow = Turn right 5 degrees

You may want to use the tips for cut and paste, and drop menus on blocks.

To call a procedure means to program a breed of agents to follow the procedure. You want the player agent to follow the Keyboard Controls procedure so you will call the procedure in the Run block (see next challenge).

Programming Tip

Windows computer:

Ctrl-C = copy and Ctrl-V = paste

Mac computer:

Cmd-C = copy and Cmd-V = paste

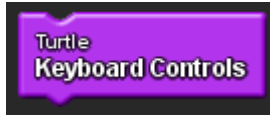
Example: Click on the orange if block to copy and paste the entire thing, including the attached blocks. Use drop menus to change blocks to the desired arrow key and associated movement block.

Challenge 4: Agents' behavior







Recall that the Forever block runs instructions from top to bottom in a forever loop. Since you only want the game to last 90 seconds, you will use a Run block.

Program the following:

- Set the seconds to 90.
- Have the player agent call the Keyboard Controls procedure. The call block is found in the My Blocks drawer that corresponds to the page where you defined the procedure. It looks like this:



- Add show clock block to the existing setup block.

Block	How it works	Drawer
	Usually placed in the Runtime page of the canvas. Works the same way as the Forever block except that it only runs for the number of seconds (secs) specified by the programmer. Each breed has its own hook for adding blocks/commands. Creates a corresponding button in the Runtime window, like Setup. When clicked, runs the instructions in a loop for each breed, over and over, until the number of seconds indicated or the user clicks the run button again. Block can be renamed and you may have more than one run block. Each instance of the block will have its own corresponding button in the Runtime area of the Spaceland window.	
	Placed in the setup block to show the clock in the Spaceland window.	
	Placed in the beginning of the setup block to reset the clock to 0.	

Test your run block by clicking the run button in the Spaceland window. Click once on Spaceland terrain to activate the keyboard controls.









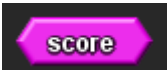

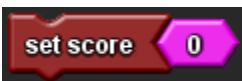







Switch Driver/Navigator roles!

Challenge 5: Interactions between agents

Use Collision blocks found in the My Blocks palette drawers to program what happens when agents collide into each other. Remember that the basic idea of the treasure game is for the player agent to collect treasure and avoid obstacles.

Program the following:

- Player – Treasure Collision: Player increases score by 1. Treasure dies.
- Player – Obstacle Collision: Player dies.
- Initialize (set) score to 0 in the existing setup block.
- Connect show score block in the existing setup block.
- To make the game harder or easier, adjust the numbers of starting obstacles and treasures

Block	How it works	Drawer
	Usually placed on the Collisions page to program instructions for agents when they “bump” into each other.	
	In a collision block, gives the ID number of the other agent.	
	Deletes the agent whose ID number is attached. For example, if you want Obstacles to kill Player, you can connect blocks kill-collidee to the Obstacles hook.	
	Deletes the agent that runs the die command. For example, if you want Player to die when it collides with Obstacles, connect die to the Player hook.	
	A variable that holds a numerical value that represents the score.	
	Used to set the score to a numerical value – either a constant (single pink number block) or a math expression that is evaluated to a single value. For example, if you wanted to increase the current score by one, you can set score to the score + 1 using the + block and number block	
	May be in used in the Setup block to display the score on the right bottom corner of the Spaceland window.	
	A math expression that evaluates the sum of the values on the left and right sides of the operator.	
	A math expression that evaluates the difference of the values on the left and right sides of the operator.	

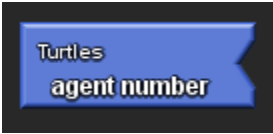



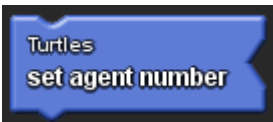

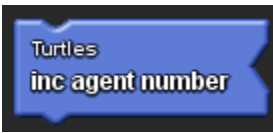



Challenge 6: “Lives” for Player agent

Instead of the Player agent dying immediately when it collides with an obstacle, you can program a “lives” (or “health”) variable for the player agent that decreases every time the player agent collides with an obstacles. When lives get to zero, the player dies. One way to program this is to use an agent variable, which functions like a trait. Every agent has its own copy of this variable, which can be seen in the agent monitor.

Programming Tip

Remember: You can rename variables when you declare them!

1. Declare an agent number variable and rename it to something meaningful like “lives” or “health” (no quotes).
2. Initialize (set) the lives variable to a starting value of your choice in the create player block located in the existing setup block.
3. Set the lives variable to decrease every time player collides with an obstacle. Program the player agent to say its lives so that the player knows how many lives are left. Also program the obstacle to die so you don’t get multiple collisions.
4. In the run block, player breed checks if its lives variable is ≤ 0 . If true, then player dies.

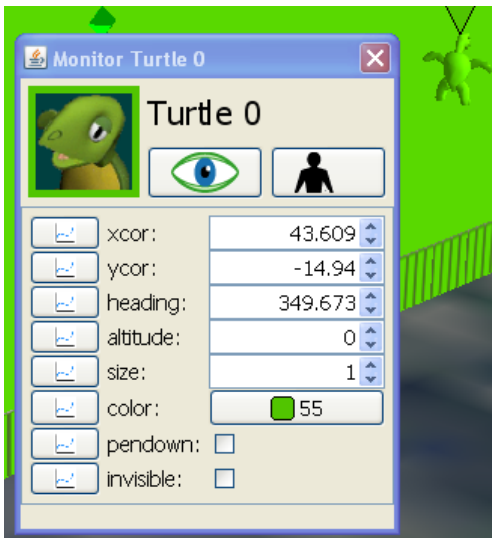
Block	How it works	Drawer
	Drag to a canvas page to declare an agent number variable. Every agent has its own copy. Can be renamed to a more meaningful name that describes its function. Generates a bunch of related blocks (see next three blocks). For example, if you drag the agent number block from the Variables drawer to the Turtles page, the related blocks will be generated in the Turtles drawer.	
	Automatically generated after you declare an agent number variable. Holds a number value. Works just like any agent trait.	 Found in the drawer with the same name as the page where the variable was declared.
	Automatically generated after you declare an agent number variable. Sets the value of the agent number variable to a numerical constant or the result of a mathematical expression. For example, if you wanted to increase the agent’s variable by one, you can set agent number to the agent number + 1 using the + block and number block	 Found in the drawer with the same name as the page where the variable was declared.
	Automatically generated after you declare a shared number variable. Increases the value of the agent number variable by some numerical constant or the result of a mathematical expression. For example, if you want to decrease the agent number variable by 1, instead of using set agent number (agent number - 1) you can just use inc agent number -1.	 Found in the drawer with the same name as the page where the variable was declared.
	When executed by an agent, a speech bubble pops up above an agent with the text indicated in the pink text block. Can also say a number block.	

Challenge 7: Adjust traits for fun

Agents come with a number of built-in traits that you can adjust. Open the traits drawer to see them all.



You can also click on an agent to open the agent monitor to see the current values of its traits.



Explore and try adjusting the traits of your agents in the Create-Do blocks. Sample code:



Programming Tip

You can place agents at a particular location using x and y coordinates, which range from -50 to +50 along both the x- and y-axes. (0,0) marks the center patch of Spaceland.

Example: To position the player to start in a corner of Spaceland, you can use (45,45), (-45,45), (-45,-45), or (45, -45).

Programming Tip

The heading indicates the direction that the agent is facing, which range from 0-359 degrees. In the overhead view, north is 0 degrees, east is 90 degrees, and so forth.

Example: If the player agent is placed at (45, 45) and you it to face the opposite corner, set heading to 225 degrees, which is southwest.

Programming Tip

To delete a block in the middle of a stack, click to select the block and then press the delete key.



Example: To replace a simple create block with a create-do block, click on the existing create block in the middle of the stack and hit the delete key, and then drag the create-do block to snap it in the same place.

Switch Driver/Navigator roles!

Challenge 8: Tweaking

Try programming some of these features or think of other things you can do to make your game more interesting!

- Program the treasure to spin.
 - Hint: Check out the movement drawer.
- Make different color treasure worth different number of points. You may want to create batches of treasure with particular colors in the existing Setup block.
 - Hint: Use create:do blocks to control the traits
- Add say or play sound during collisions.
- Program an ending for the game if the player wins (passes some threshold for score) or loses.
 - Hint: Make a procedure that runs when the threshold is passed

Block	How it works	Drawer
	Agent plays a sound file specified by the “sound” block.	

Programming Tip

Typeblocking is a feature in which you can just start typing the name of the block you want and press enter to make the block fly into the canvas or select it from a drop menu. If you’ve selected an existing block or stack, the new block will connect if possible.

Example: To construct the expression `set score (score + 1)`, type `set score` and press enter. Type the `+` symbol and enter. Type `score` and enter. Type `1` and enter.





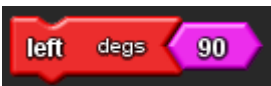





Challenge 9: Rival agents

To increase the excitement of playing your game, add rival agents who can also collect treasure, thus reducing the number available for the player.

1. Add the rival breed.
2. Create one or more rival agents.
3. Program the rival agents to move. Start with something simple at first, like forward 1.
4. Program the collision between rival agents and treasures so that treasures die.

Improvements:

- Try different movement instructions to see which will allow the rival agent(s) to cover more ground.
- Adjust the number of rival agents until you achieve the right level of challenge.
- Try adding randomness to your movement instructions.

Block	How it works	Drawer
	Agent(s) moves forward the number of steps according to the pink block.	
	Agent(s) moves back the number of steps according to the pink block.	
	Agent(s) moves left the number of degrees according to the pink block.	
	Agent(s) moves right the number of degrees according to the pink block.	
	Returns a whole number value between 1 and the number indicated in the pink block. Usually used in a math expression or with a movement block.	

Programming Tip

Why do the collisions seem to not work sometimes?

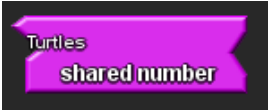

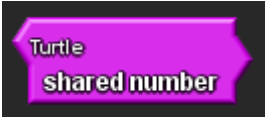

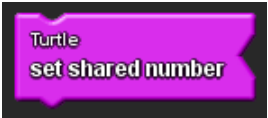

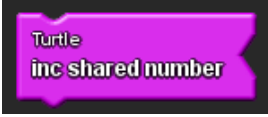



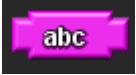

If you program the rivals to move more than 1 step at a time, it's possible that the agents will go through another agent instead of colliding with it.





Switch Driver/Navigator roles!

Challenge 10: Rival agents have their own score

Program the rival agents keep their own score so that the player's goal is to beat the rival agents' score. This involves creating a variable that functions like score. It's a little complicated so study the table of blocks below.

1. Declare a shared number variable and rename it to something like "rival score" (no quotes).
2. Initialize (set) it to 0 in the existing setup block.
3. Set it to increase in the appropriate collision block.
4. To see the value of the rival's score in the Spaceland window, use the set status block and convert the rival's score to text. See table below for these and other text drawer blocks.

Block	How it works	Drawer
	Declares a shared number variable when dragged to a canvas page. There is only one copy and it can be modified by any agent . Can be renamed to a more meaningful name that describes its function. Generates the next three blocks with the title of the page it is declared on. For example, if you drag the shared number block from the Variables drawer to the Turtles page, the related blocks will be generated in the Turtles drawer of the My Blocks palette.	
	Automatically generated after you declare a shared number variable. This block holds a number value, just like the score block. Use this block whenever you want to access the number represented by the shared number variable. It can be used in any procedure, forever loop, run loop, etc. You use this to say "what number is currently represented by this variable?"	 Found in the drawer with the same name as the page where variable was declared.
	Automatically generated after you declare a shared number variable. Sets the value of the shared number variable to a numerical constant or the result of a mathematical expression.	 Found in the drawer with the same name as the page where variable was declared.
	Automatically generated after you declare a shared number variable. Increases the value of the shared number variable by some numerical constant or the result of a mathematical expression. For example, if you want to increase the shared number variable by 1, instead of using set shared number (shared number + 1) you can just use inc shared number 1 .	 Found in the drawer with the same name as the page where variable was declared.
	The status bar is located on the lower left side of the Spaceland window, in the same row as the score. It can take a text argument (see next block (abc) (note the right angle shape of the socket).	
	A text argument. Highlight the abc text and type to replace it with your own text.	

	Converts a number block into text. For example, you can plug in a shared number variable on the right side to change it to text for set status to display.	
	Concatenates (joins) two text arguments. For example, you can put a regular text argument on the left side and a “to text” number variable on the right side to join them into a single text argument.	

Sample Code:

One way you can set up the **set status** block and turn a number into text with the **to text** block.

