Fountain Programming Activity

Fountain Programming Activity

1

Setup

Prerequisites: Orientation, Epidemic or Paintball

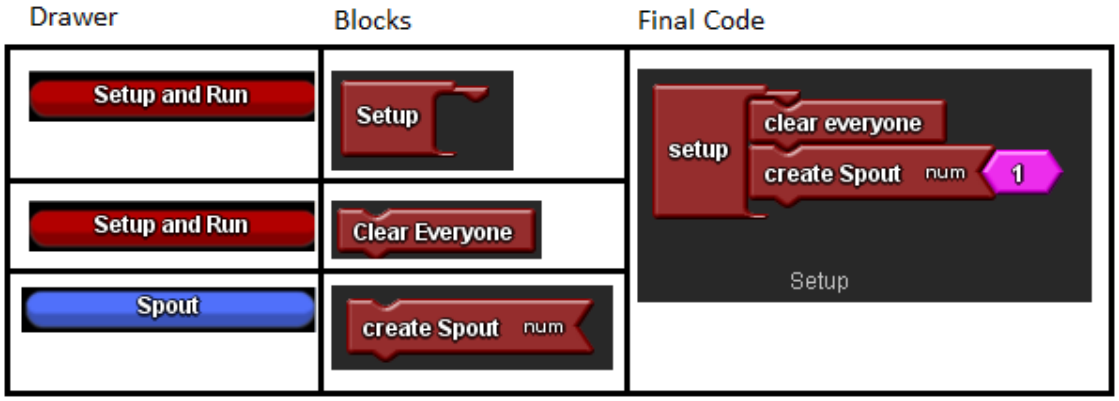
1.1 Create Your Breeds

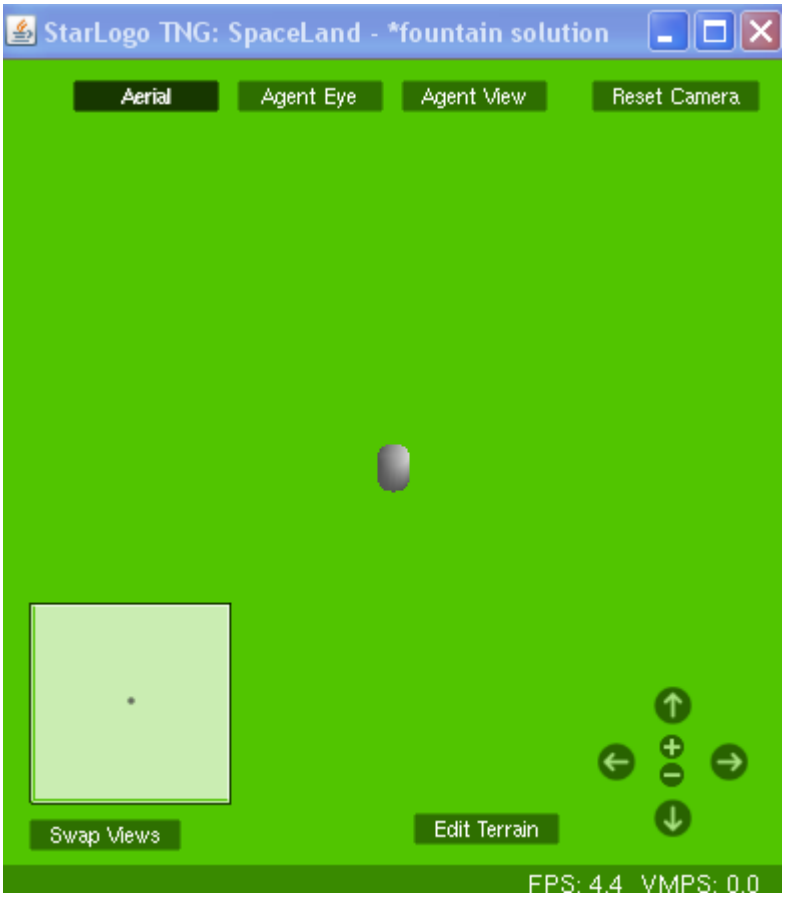
In the "Edit Breeds" window, create 2 new breeds.
Select any 3 different "basic-shapes" for the breeds and rename them to "water", "droplets", and "spout".
It is important that you use this order: Water, Droplets, Spout

1.2 Set Up Spaceland

The "setup", "Clear Everyone", and "create Spout" blocks can be found in the corresponding drawers. The end result will clear spaceland and create 1 spout in the center by default.

Test out your code by clicking the **setup** button that appeared below Spaceland!

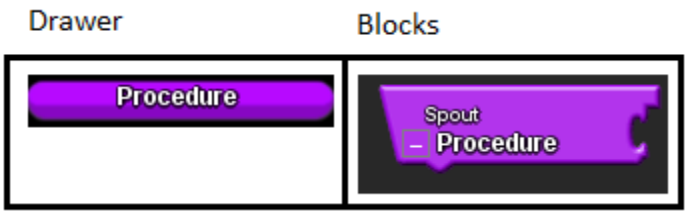

Drawer
Blocks
Final Code
Setup and Run
Setup
setup
clear everyone
create Spout num 1
Setup and Run
Clear Everyone
Setup
Spout
create Spout num


StarLogo TNG: SpaceLand - *fountain solution
Aerial
Agent Eye
Agent View
Reset Camera
Swap Views
Edit Terrain
FPS: 4.4 VMPS: 0.0

2.1 Create the Launch Procedure

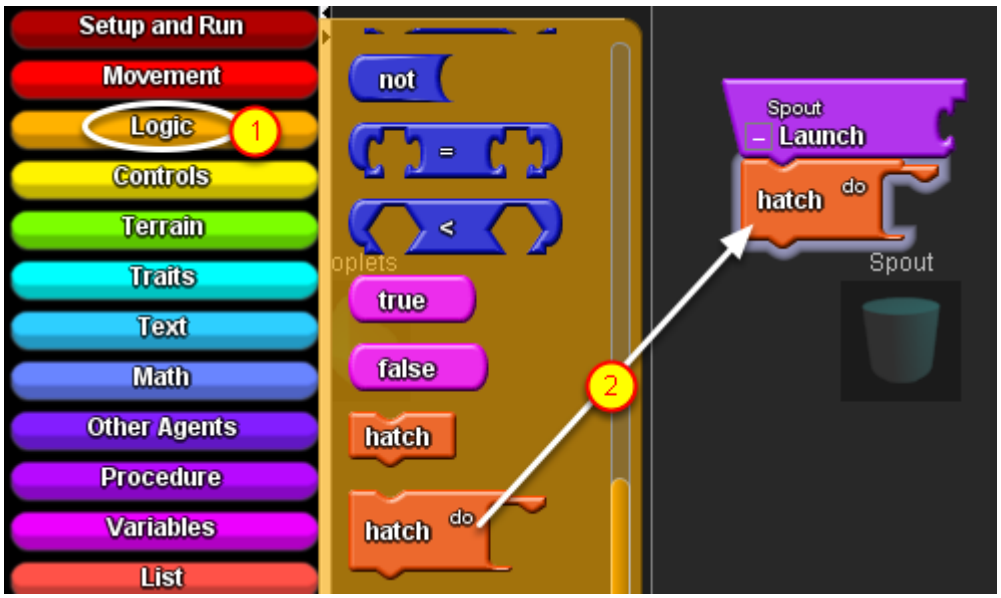
Create a Procedure on the "Spout" page. Rename it to "Launch".
Recall that procedures are useful when you want to use a set of instructions multiple times or when you want to break down a complicated task into discrete parts and program each part separately in an easy-to-read way.
There are two important 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)



2.2 Use the "Hatch:do" Block

In the "Logic" drawer, connect the **hatch:do** block.
*The **hatch:do** block creates a copy of the agent and then the copy completes the set of code in the block.

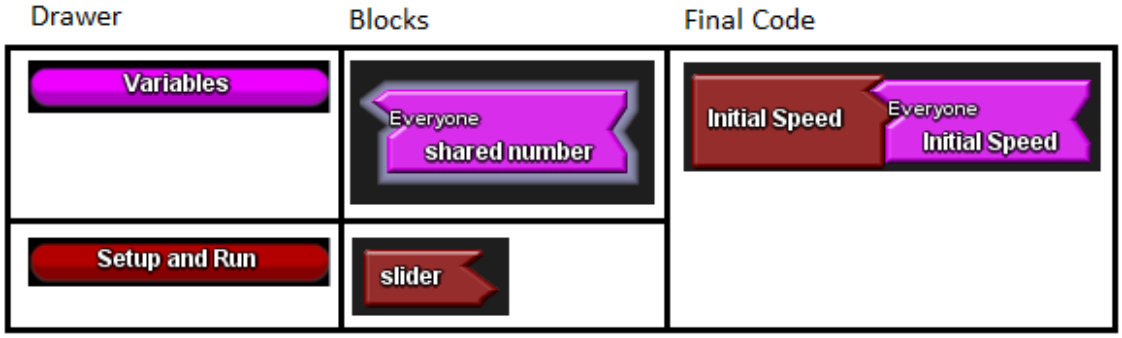


2.3 Use the "Agent Number" Block

We need to make the agents have a vertical velocity variable.
On the "Everyone" page of the canvas, go into the "Variables" drawer and pull out the **agent number** block. Make sure it's a BLUE block. Rename it to "Z Velocity".
*The blue color tells you that each individual agent of a breed will its own variable for the vertical velocity.

2.4 Create a Slider to Control the Initial Speed

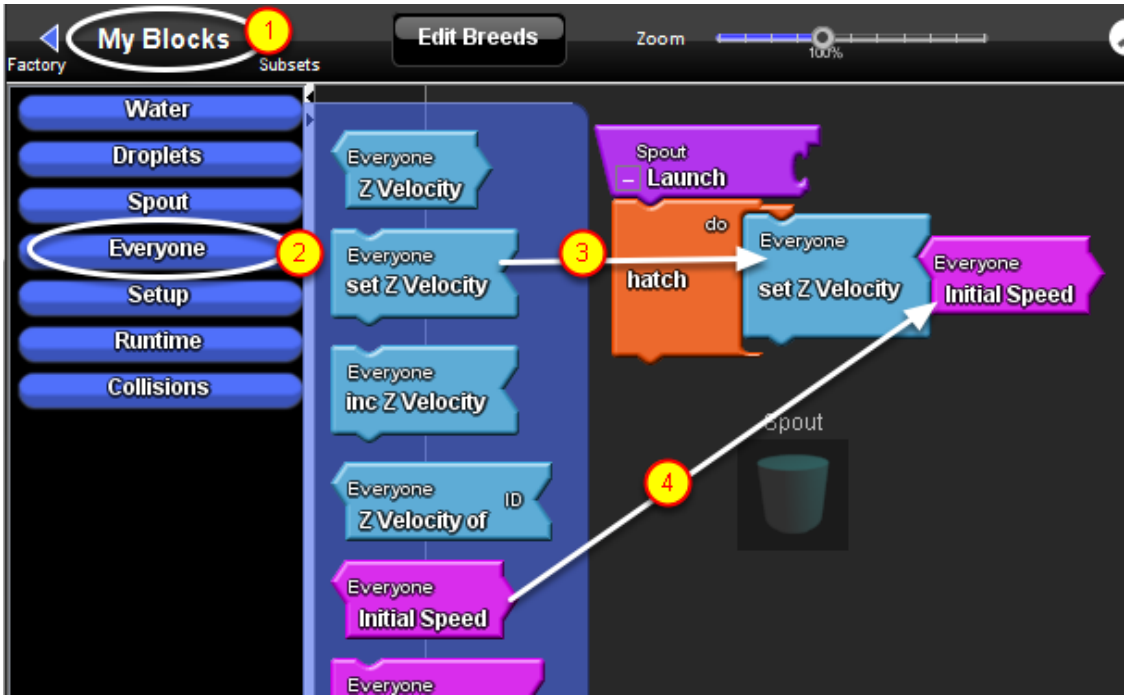
On the same page called "Everyone", create a **shared number** variable. Make sure that the block is PINK. Snap in the **Slider** block to be able to adjust the initial speed from the Runtime area of Spaceland..
*The pink color "shared number" block means there is only one copy of that variable that all agents can access and change.



2.5 Set the Initial Velocity

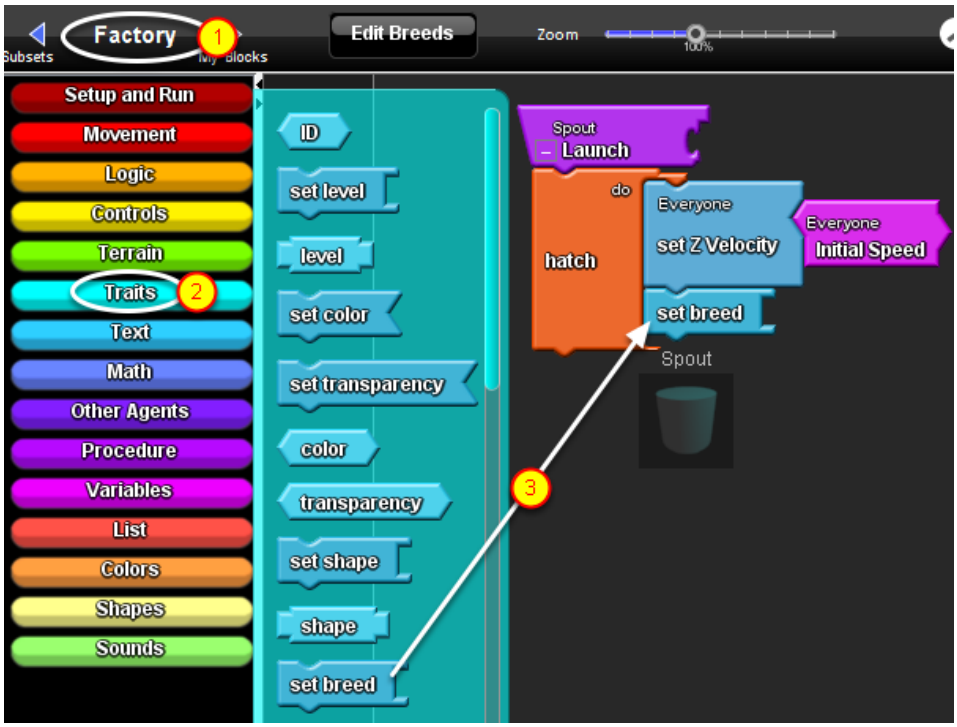
Scroll left on the canvas pages until you find the Launch procedure on the Spout page.
Click on the arrow above the drawers to switch to the "My Blocks" palette, open the "Everyone" drawer, and drag/drop the "set Z velocity" and "Initial Speed" blocks into the Hatch:do blcok. This

will set the initial velocity of the newly hatched agent to the value you choose on the slider.
*This means that when the spout hatches a new agent, the agent (soon to be water droplet) will have a velocity equal to the inital speed according to the slider adjusted by the user

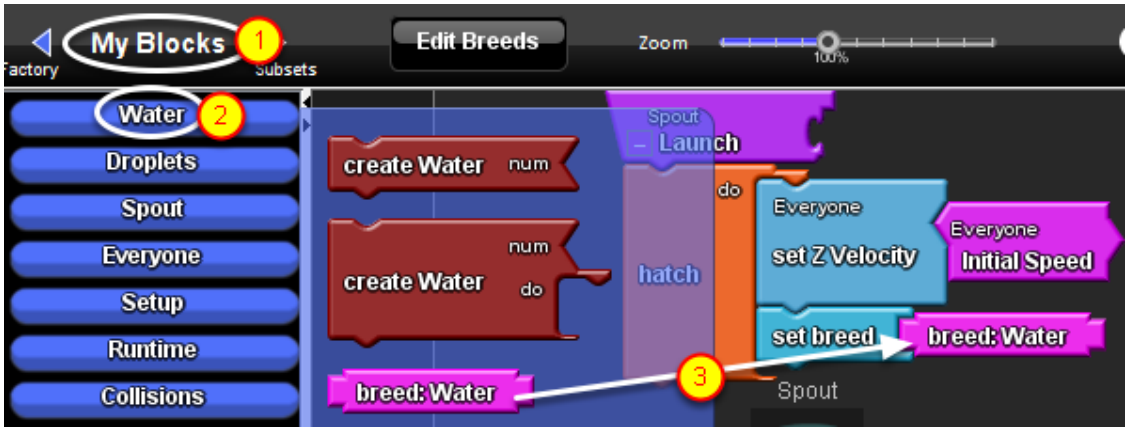


2.6 Set the Breed to Water

The new spout that was hatched needs to become the "water" breed.
Back in the "Factory" palette, open the "Traits" drawer and take out the **set breed** block.

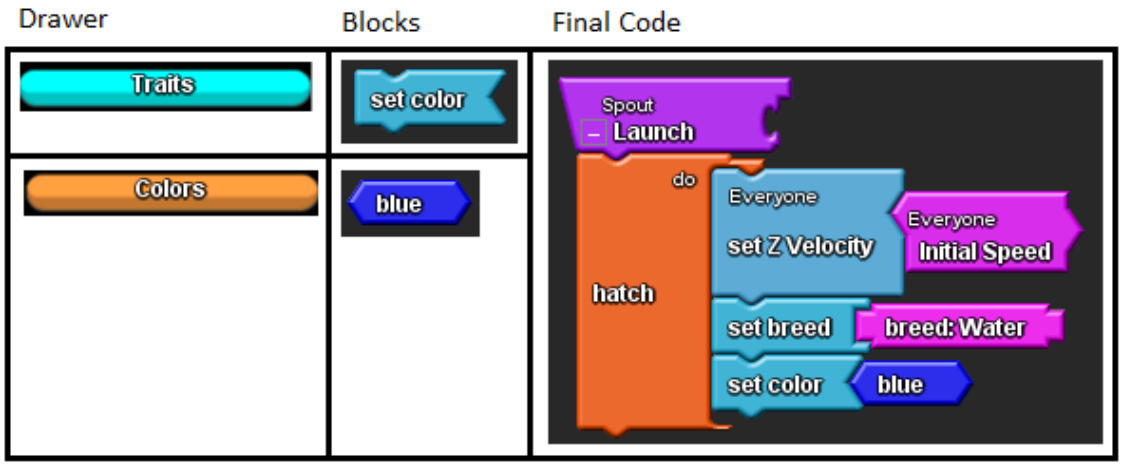


Back in the "My Blocks" drawer, open the "Water" drawer and snap in a **breed: Water** block.



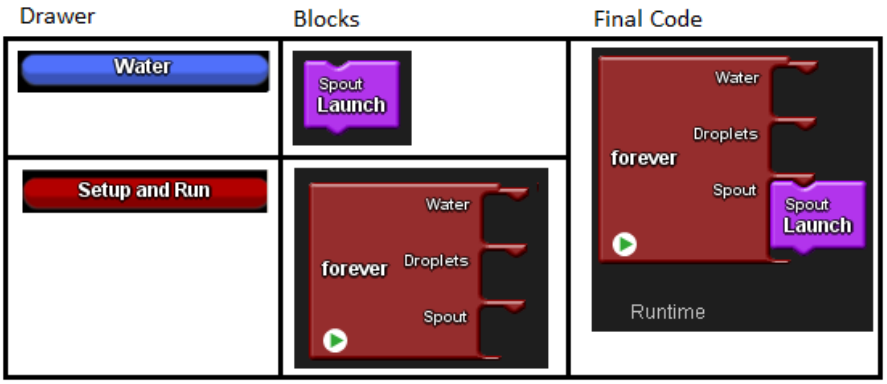
2.7 Set the Color

Using the **set color** and **blue** blocks, turn the newly created "Water" agent blue.



2.8 Call the Launch Procedure

Scroll to the Runtime page. Using the **Forever** and **Launch** blocks, make the "Spout" agent create "Water" agents.

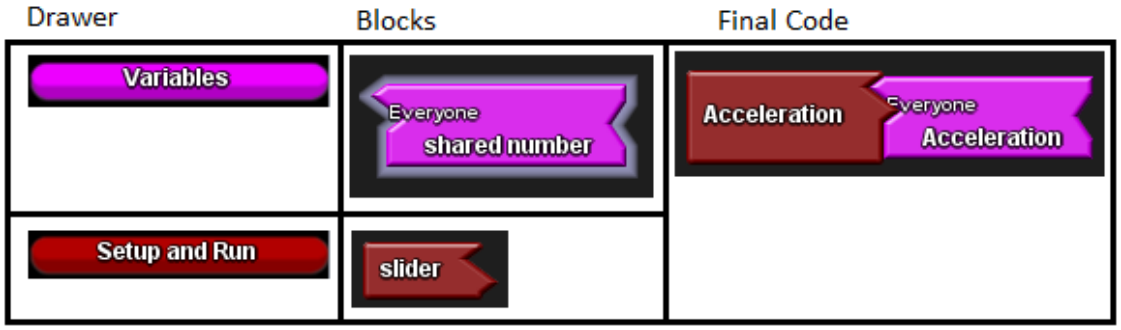


After all of this, Spaceland should appear like this when the "forever" button is running.

***IMPORTANT NOTE: Click the forever button again to stop the simulation. If it continues for a long time, the model will create about 4000 water agents and the speed will slow down significantly. Later in the code, we will make sure the delete the agents that go off the screen.**

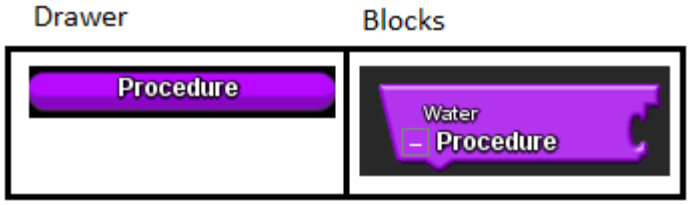
3.1 Create the Acceleration Slider

On the "Everyone" page, create another **Shared Number** connected to a **Slider** and call both "Acceleration". This will let you adjust the acceleration in the model. Think about acceleration as gravity.



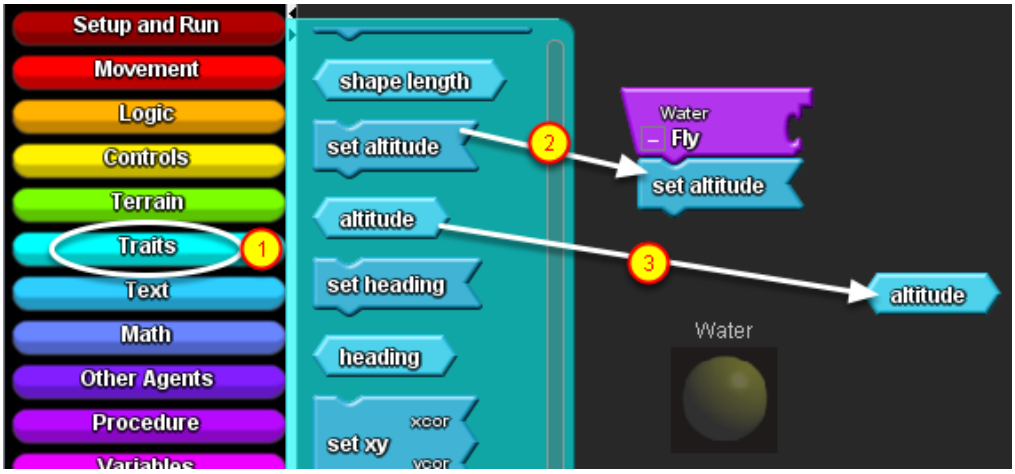
3.2 Initialize the Fly Procedure

On the "Water" page, create a procedure and call it "Fly"

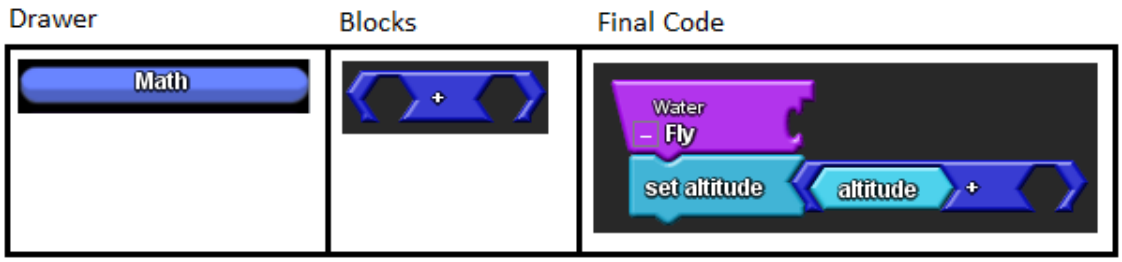


3.3 Set the Altitude

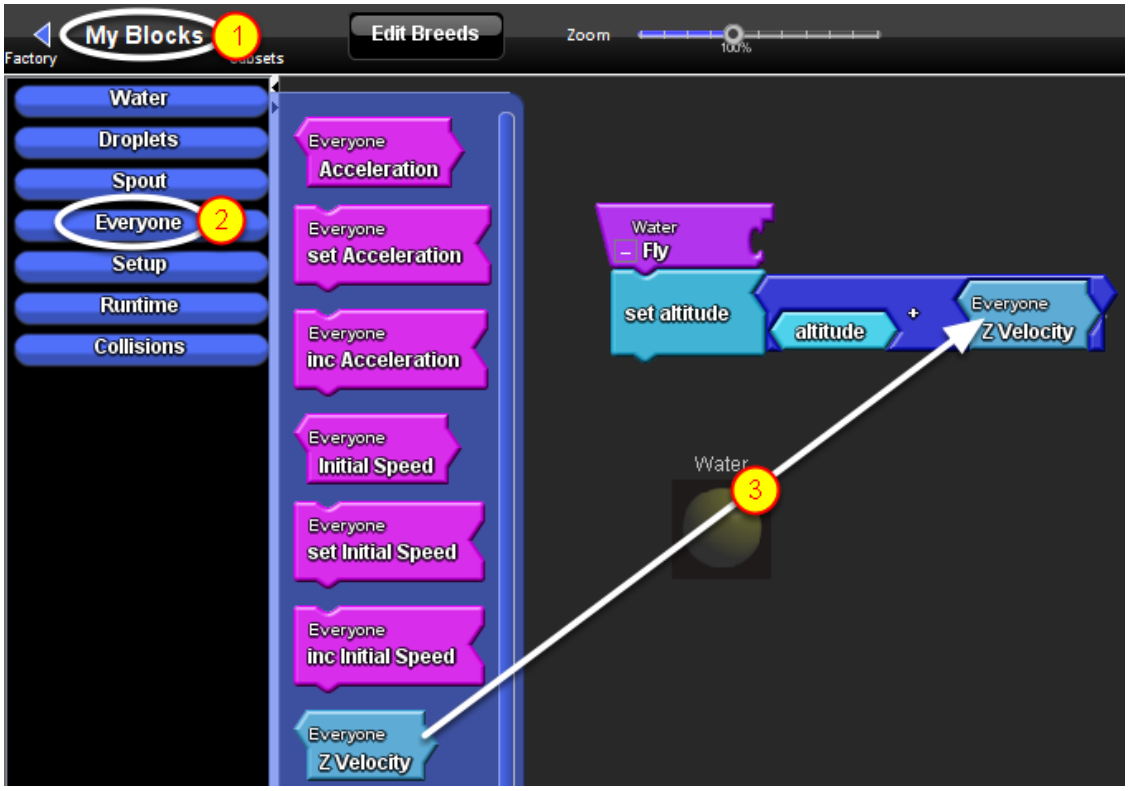
From the "Traits" drawer, drag out the **set altitude** block and an **altitude** block. We will control the vertical position by adding the Z velocity to the current altitude.



From the "Math" Drawer, drag out the **+** block. Snap it into the **set altitude** block and put the **altitude** block into the first slot.
This means that everytime the **Fly** procedure is run by an agent, it will set its altitude to its current altitude plus a number.



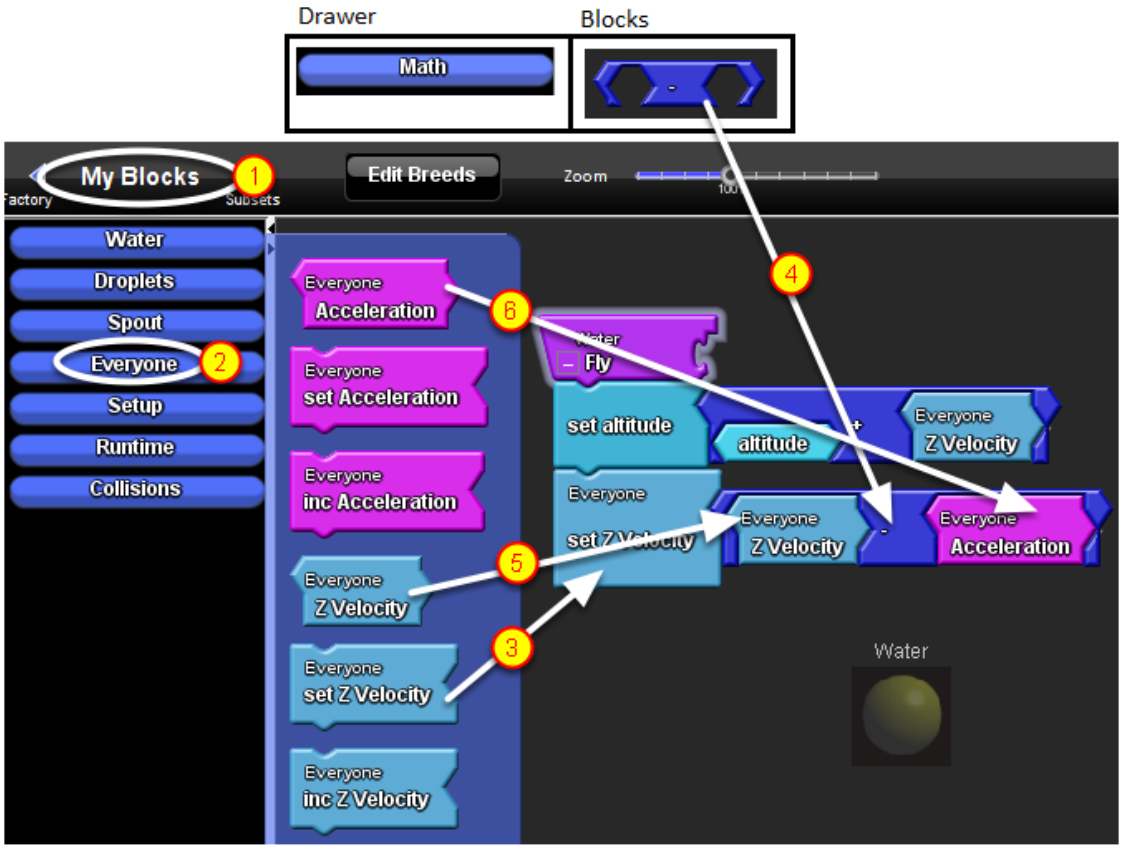
In the "My Blocks" palette, open the "Everyone" drawer and drag/drop the **Z Velocity** block into the 2nd slot of the **+** block. The Z Velocity block holds the number of whatever the variable was set to last. Basically, this tells the water agent to set its height location (altitude) to its current altitude plus its current velocity.



3.4 Accelerate the Vertical Velocity

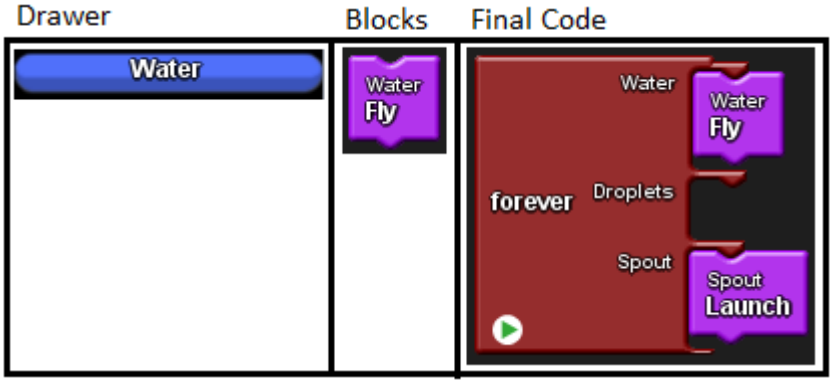
In the "My Blocks" palette, open the "Everyone" drawer and drag/drop the **set Z Velocity** block under the set altitude block.
Snap the **-** block into the **set Z Velocity** block and then put in the **Z velocity** and **Acceleration** blocks. The **Acceleration** block is the number value that the slider is set to.

After this command, the **Z velocity** will be its previous value minus the acceleration. You can imagine this as the water agent moves up in the air, its velocity will decrease as gravity (accleration) acts on the water. Eventually, Z Velocity will become negative. A negative Z Velocity means that the atitude will decrease as the water agent "falls" back down.



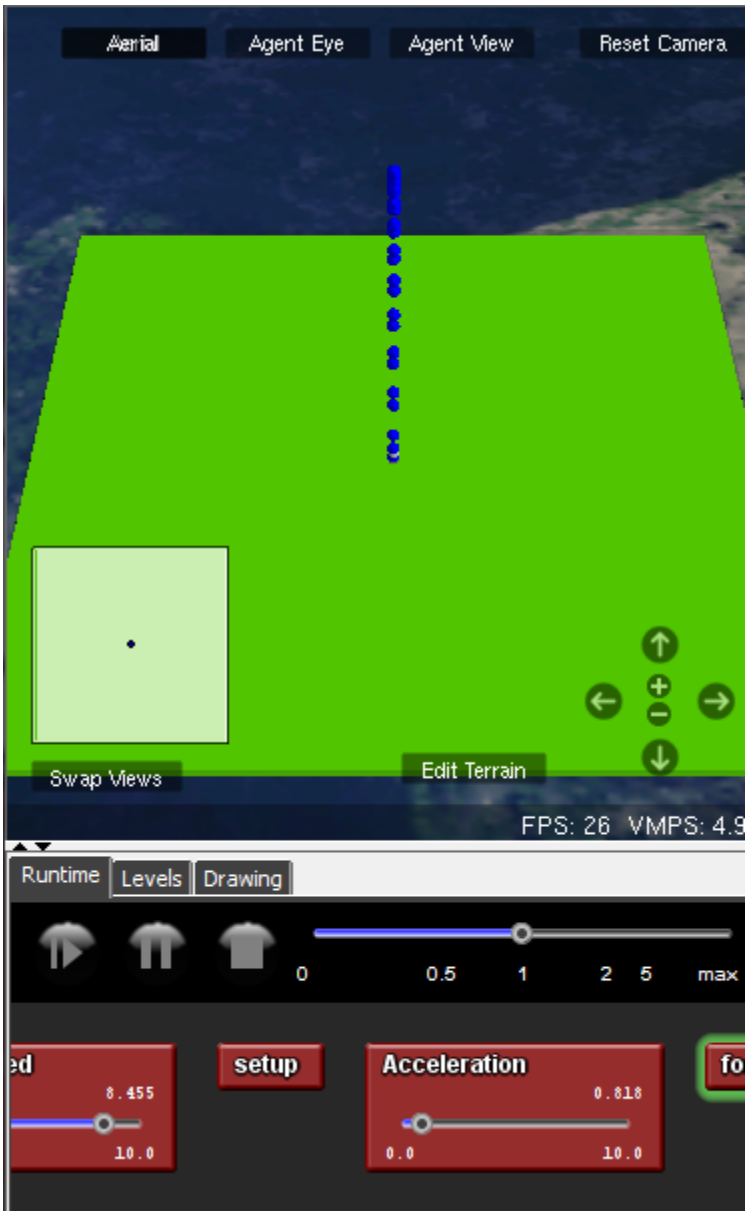
3.5 Call the Fly Procedure

Find the **forever** block on the runtime page of the canvas. Go to the My Blocks palette, open the Water drawer, drag out the **Fly** procedure block and snap it on the Water hook.



In the Spaceland window, adjust the Initial Speed and Acceleration sliders. Clicking on the "forever" button should make the water fly up and fall down. Adjust the Accleration to see the effect of low and high "gravity" and low and high "initial speed".

***IMPORTANT NOTE: Click the forever button again to stop the simulation! (For the same reasons as in step 2.8)**

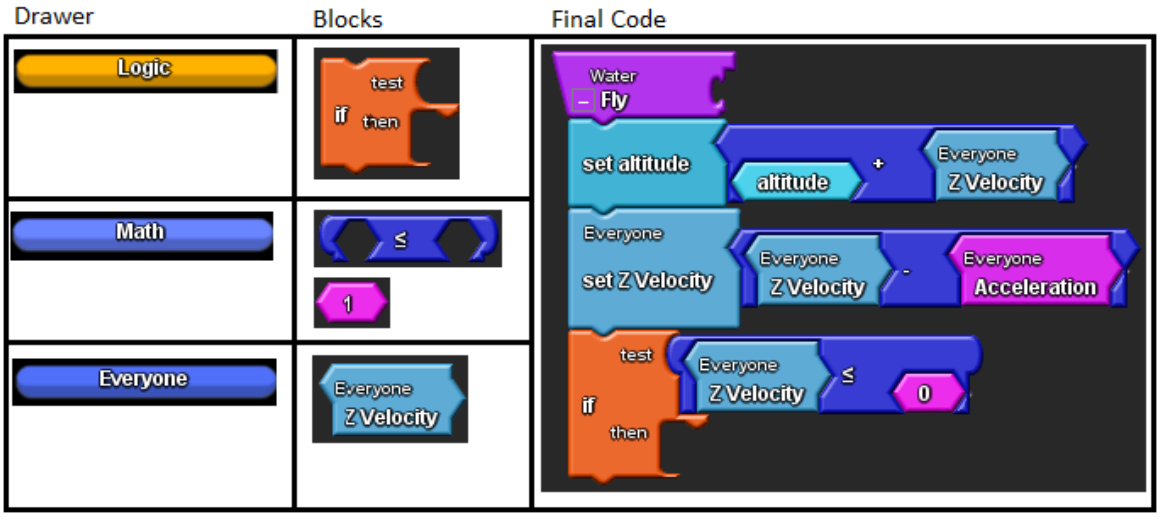


Transforming the Water into Droplets

So far, the water flies up and then falls down as any object should. In most fountains, the water separates at its peak. In this section, you will program the water agents to test when they reach the top, and when they do, to program the water to change its breed to droplets, hatch many copies of these droplets to that the droplets make a nice ring around the spout, ready to follow the droplets code (which will be programmed in the next section).

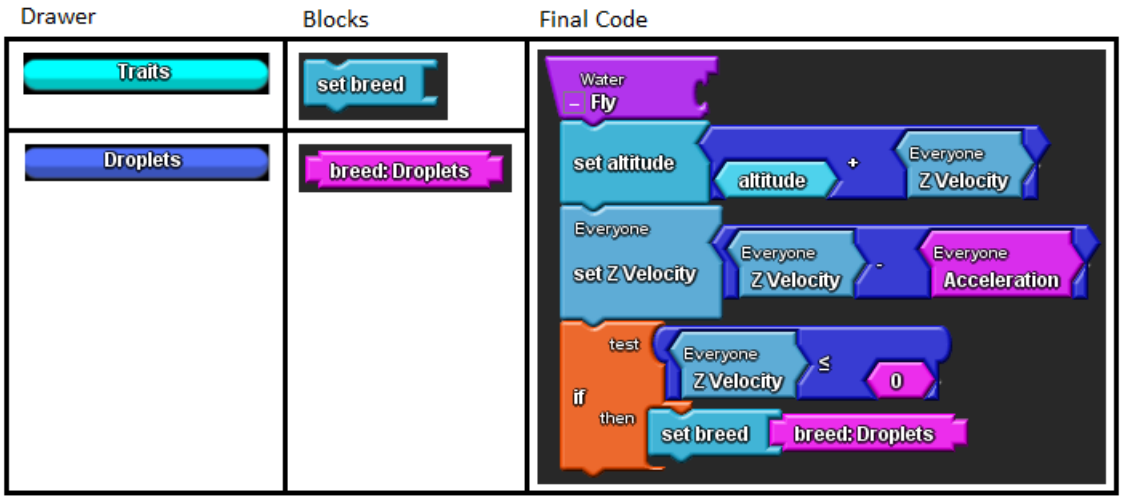
4.1 Test for the Peak of Flight

You can use the velocity to test if it has reached the top. Use the **If**, **< or =**, **number**, and **Z Velocity** block to test if the velocity is lsetess than or equal to 0.



4.2 Change the Breed

Change the "Water" agent into the "Droplet" breed when the **Z Velocity** is less than or equal to 0.

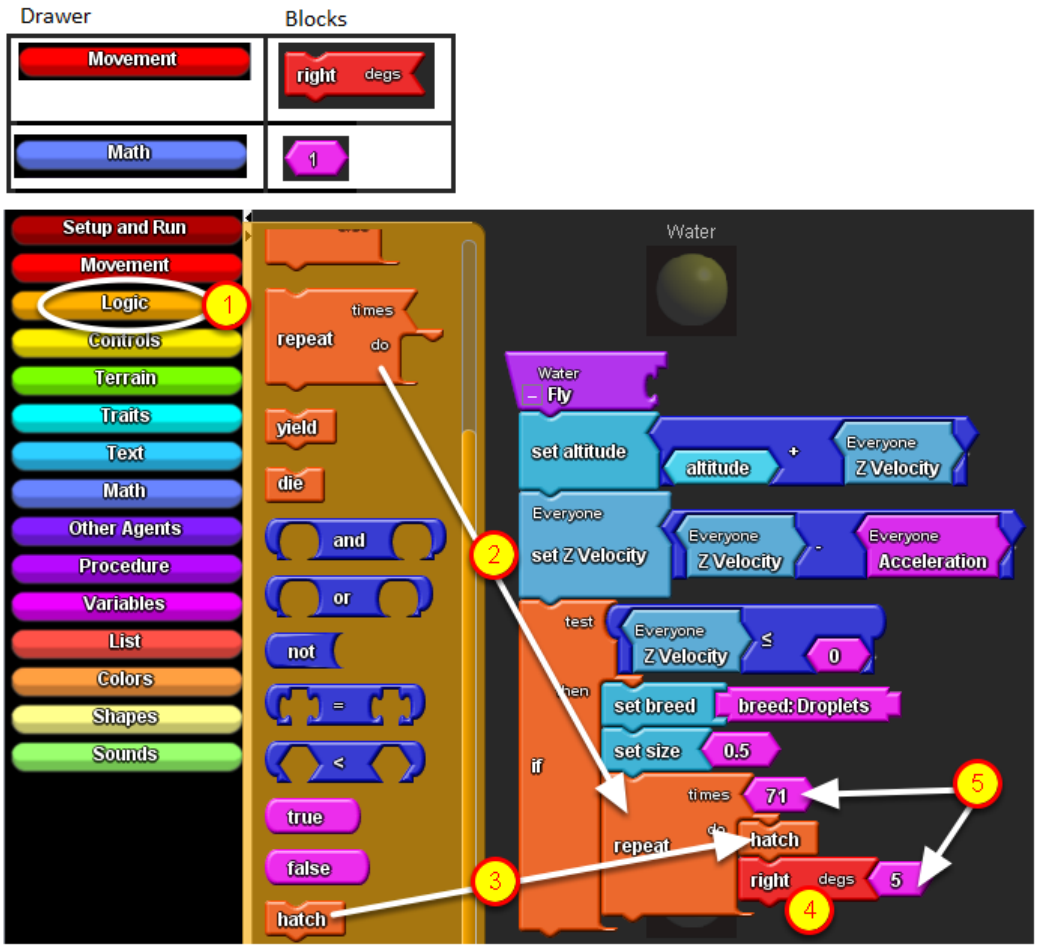


4.3 Shrink the Droplet

Make the droplet smaller.
In the "Traits" drawer, drag out the **set size** block and snap it under the set breed block. Change the **1** block to 0.5

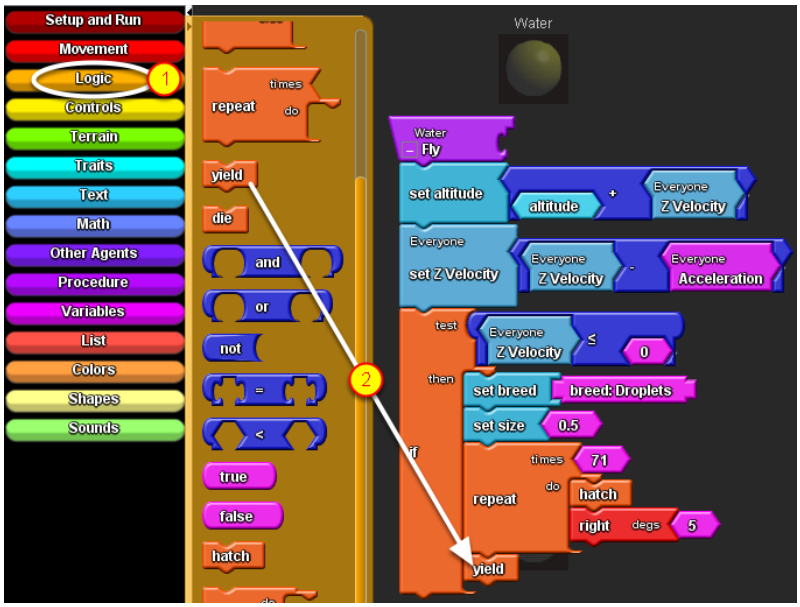
4.4 Hatch New Droplets Facing Out

Now that the "Water" agent is now a "Droplet" agent, we want the new "Droplet" agent to hatch more "Droplet" agents, facing outward in directions that are equally spaced around a circle (360 degrees).
From the "Logic" drawer , take out the **repeat** block. Put a **hatch** and **right** block into the "do" socket of the **repeat** block.
You ask the original droplet to turn a little before hatching so that each droplet is facing a different direction, sequentially.
The number of degrees you set the turn will define how many times it needs to be repeated. To figure out how many times it should be repeated, divide 360 degrees by the degrees of turning, and then subract 1. We subtract 1 because if you do not, then you will end up with an extra droplet in the exact same place as the first droplet.
Here, since the agent is turning right 5 degrees, # of repeats = 360/5 -1 = 71. So the original droplet agent will hatch 71 droplets.



4.5 Yield

Open up the "Logic" drawer and put a **yield** block after the **repeat** block. This will make the new droplet agent wait before it runs its next code. A more detailed explanation of the **yield** block is in Step 7.



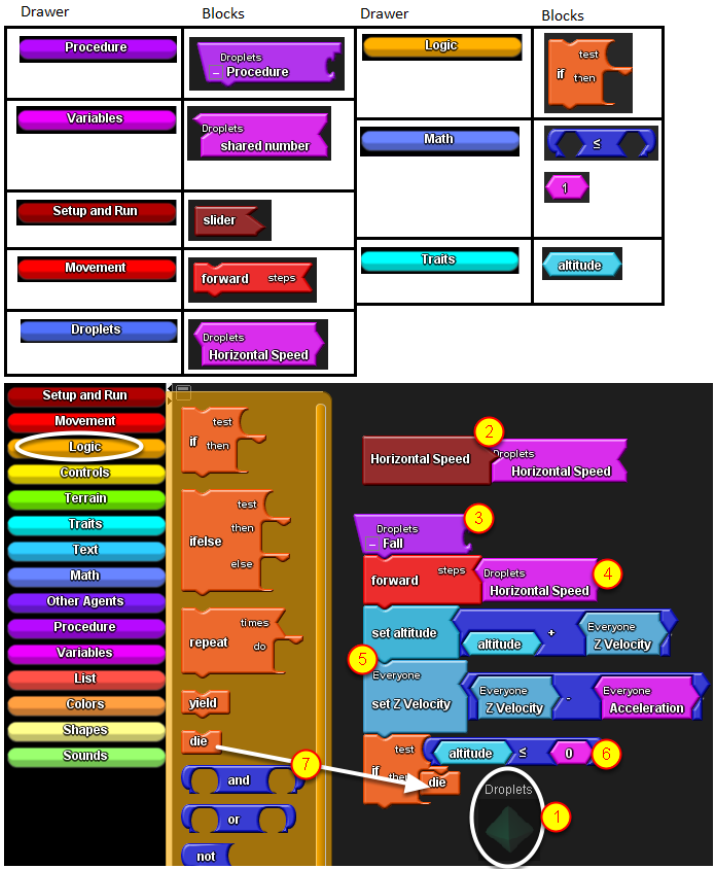
Making Droplets Fall

In the previous step, you programmed 72 droplet agents to appear. In this section, you will program them to fall in an expanding circular pattern.

5.1 Complete the Fall Procedure

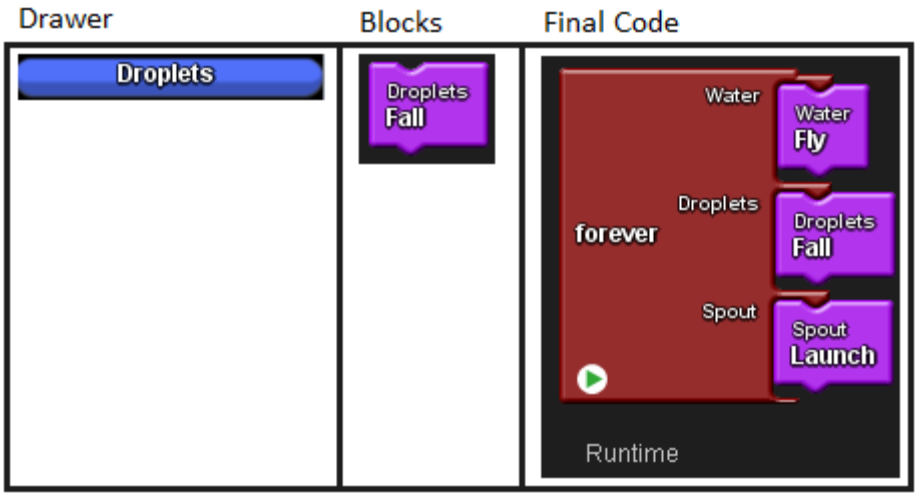
You will program the droplets to fall in the same way as the water but also move forward so that the circle expands as the droplets fall. To do this you will declare a new variable called Horizontal Speed that will indicate how far the droplets move forward during each time step. Note that the horizontal and the vertical movements are simulataneous but independent. Also, you will test to see when the droplets go below the green plane and when they do, they will die.

1. Go to the "Droplets" page of the canvas.
2. Create a slider with another **shared number** and call it "Horizontal Speed"
3. Create a **Procedure** and call it "Fall"
4. Attach the **forward** block to the **Fall** procedure and snap in the **Horizontal Speed** block from the "Droplets" drawer.
5. Copy the **set altitude** and **inc Z Velocity** sets of blocks from the **Fly** procedure
6. Make the code test the altitude by inserting an **if** block and checking if the **altitude** is **less than or equal to 0**.
7. Open the "Logic" drawer and pu the **die** block into the "then" socket of the **if** block. The **die** block deletes the agent from the simulation.

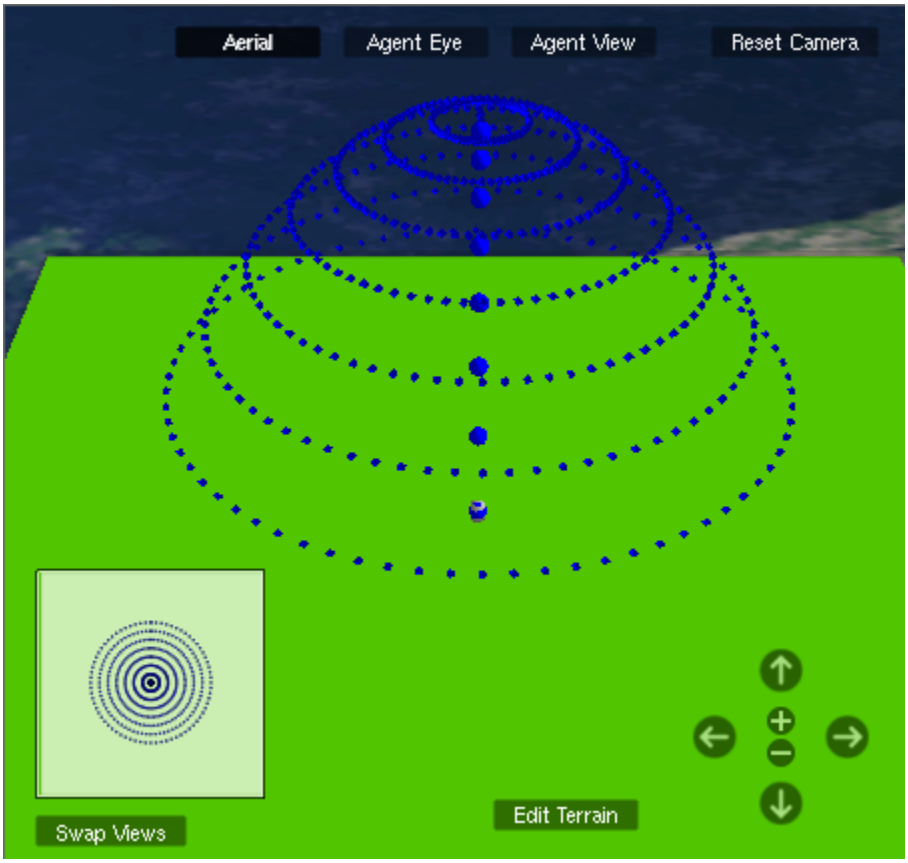


5.2 Call the Fall Procedure

To make the "Droplets" agents follow the instructions in the procedure **Fall**, take the **Fall** block from the "Droplets" drawer and connect it to the Droplets hook in the **forever** block.

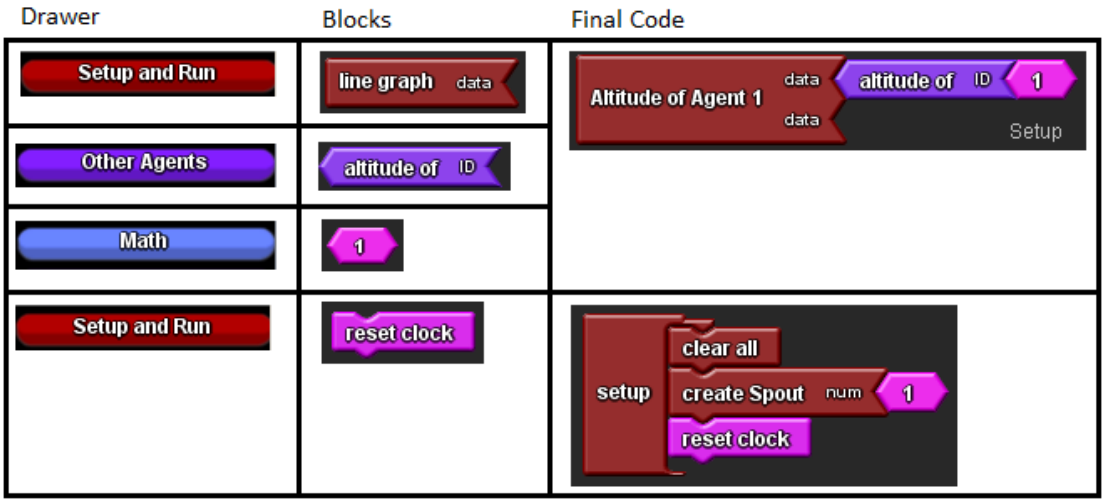


If your code works - you should have a fountain that looks like this. You can adjust the various sliders to see how they affect the behavior of the droplets.

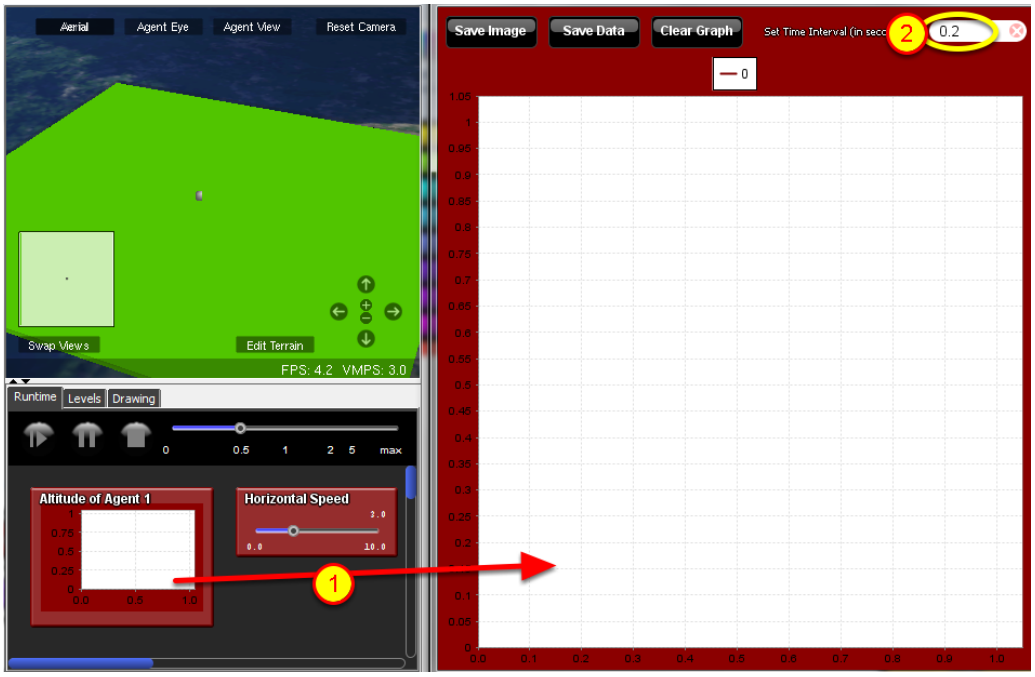


Observe the Altitude of an Agent

Create a line graph monitoring the altitude of an agent. Try monitoring more than one agent's altitude! Put the **reset** block in the **setup** block so that you get a new graph every time you click the setup button.

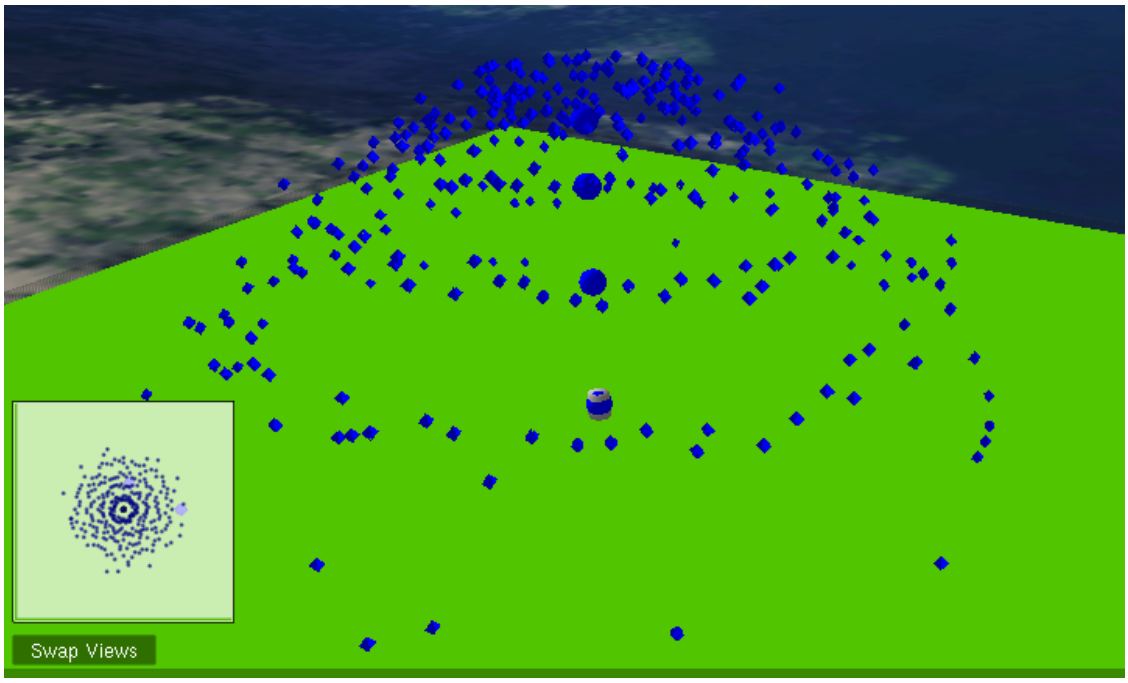


The line graph widget will be created below spaceland. Click on it and you will see a large version of the graph. make sure the Time interval is small like 0.2.
What do you notice about the shape of the graph? What does the shape tell you about how the position of the water changed as it left the spout and then came down as a droplet? Why does the graph keep repeating?



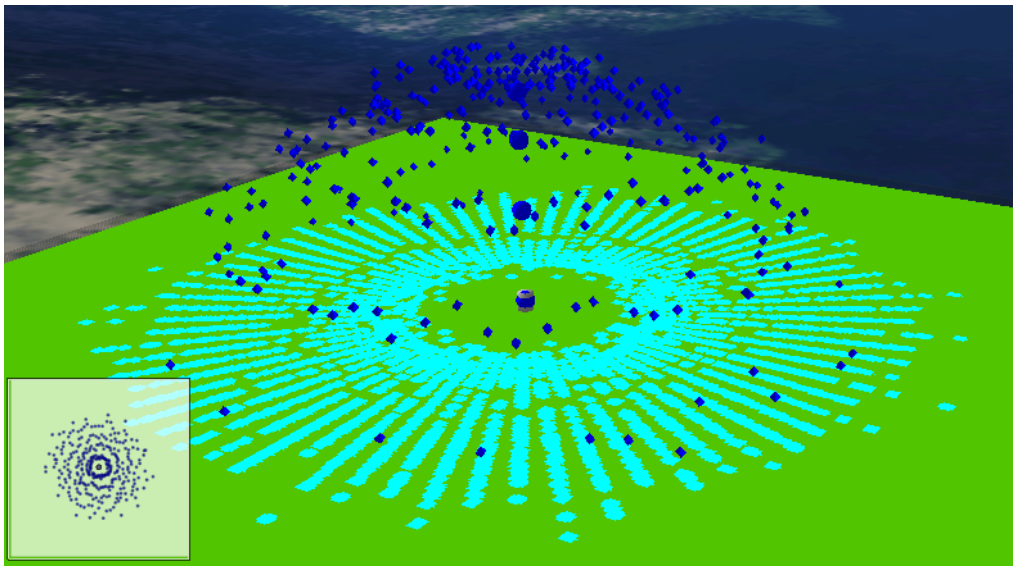
Adding randomness to falling droplets

Real fountain droplets don't spread out in a perfect circle. How can you add some randomness so that the fountain looks more the above? First try adding **random** to the Horizontal movement in the Droplets:Fall procedure. And then try adding **random** to the initial vertical placement of the droplets (when they are newly hatched - replace **hatch** with **hatch:do**) in the Water:Fly procedure.



Marking the terrain when water hits it

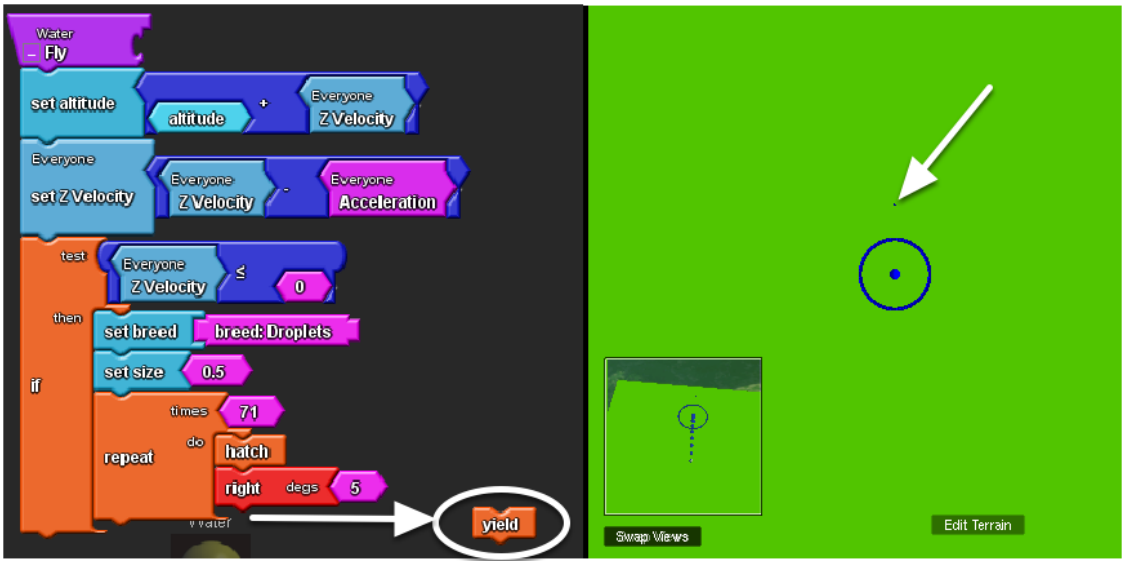
To show water hitting the ground, you can tell the droplets to **stamp** the terrain a certain color before it dies. You should also change the **clear everyone** block in setup to **clear all**, which would repaint the terrain to green when you click setup.



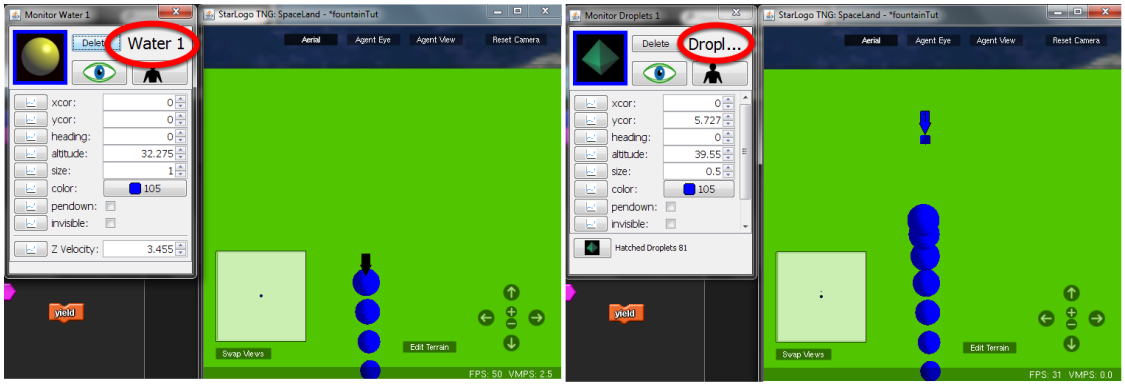
Why Yield?

7.1 What happens when Yield is removed

If you take the **yield** block out of the **if** block, the first "Water" agent that you created will run the **Fall** procedure before all of the other Droplets do.



The first "Water" agent, "Water 1" turns into "Droplet 1", hatches 71 other "Droplet" agents, and then it **Falls** before all of the other droplets do.



7.2 Why does this happen?

The computer has an internal clock that ticks. TNG can make 5 ticks every human second. The computer runs every forever loop once during each computer "tick". The computer runs the forever block in order from top to bottom. As a result, when the first "Water" agent runs the **Fly** procedure, it becomes a "Droplet" agent during the procedure. Once it finishes the **Fly** procedure it was already running, it can immediately run the **Fall** procedure when the **Forever** block moves on to the "Droplets" code. However, if you put yield in the **Fly** procedure, the transformed droplet agent is

forced to wait for the next tick to run its next code.

In summary, "yield" forces the agent to wait until the next tick before it can continue its next command, regardless of where it is in the code.