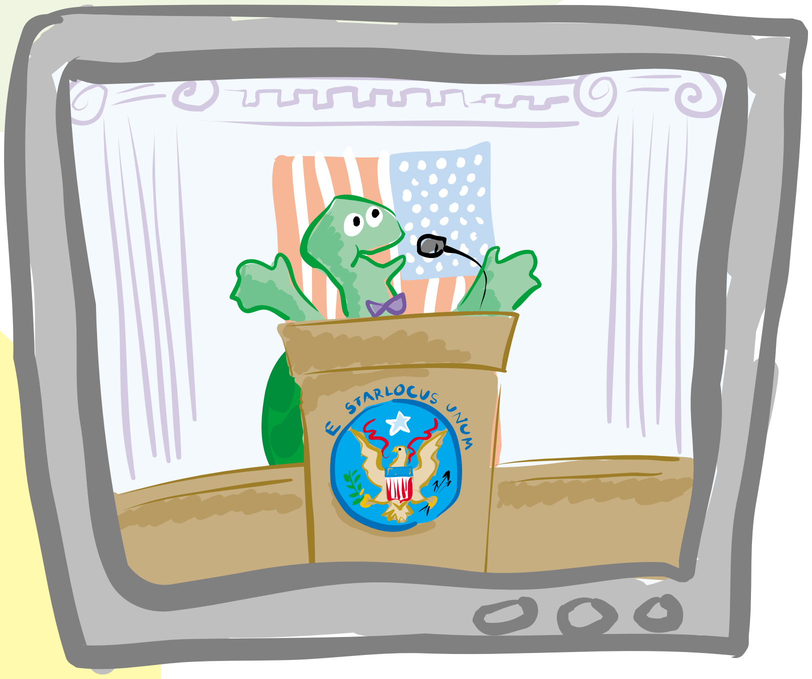
CHALLENGE 5

State of the Turtles

In the previous Challenges, you altered several turtle properties (e.g., **heading**, **color**, etc.). These properties, called turtle variables or *states*, allow the turtles to remember information. Turtles can remember additional information about themselves if you add new turtle variables.

Can you create a StarLogo project with a turtle **energy** variable? How will your turtles gain and lose **energy**? How might your turtles' behaviors or looks change as their **energy** levels increase or decrease? How does the turtles' behavior help you visualize their changing states? What assumptions do you make about a turtle's states based on its behavior? Are your assumptions accurate?

For hints on getting started, check out the sample projects ***Energizer Turtles*** and ***Rabbits***.



POSSIBLE EXPLORATIONS

- Explore different ways that **energy** can affect the speed and heading of the turtles.
- Try having more than one factor (e.g., colored patches, x-coordinate, **who** number) influence the turtles' **energy**.
- See what happens if turtles with different **energy** levels affect the environment in different ways.
- Experiment with turtles hatching other turtles when their **energy** reaches a threshold level.
- Think about how the turtle behavior that you observe gives you clues about the underlying turtle states. What assumptions do you make about the turtle's states based on the observed behavior? How might you change your project to make the correlation between underlying states and overt behavior more instructive? For example, you could make turtles with more **energy** move faster.
- How can you use variables in your model to make individual behaviors seem more lifelike?

CHALLENGE PHILOSOPHY

Every January, the President of the United States delivers the State of the Union Address in which he describes the status of the country. Typically, he addresses a number of different topics, including the budget, crime, education, health care, and taxes. He provides information on how well the country is performing in each of these areas. For instance, in 1999 crime was falling to record lows while health care costs were increasing. The information he gives about each topic helps his constituents construct informed opinions about the status of the country. Just as the President describes the condition of the country, the status of other systems can also be described or assessed by looking at the underlying components. For example, your doctor weighs you, takes your temperature and blood pressure, and measures your cholesterol levels. Collecting information about each of these states helps her create a picture of your general health and well-being.

Often without even being conscious of it, you think about the states of organisms or whole systems. As you observe behaviors in the world, you might make guesses about the reasons underlying these behaviors. Think about the last time you were "people watching." If you saw a woman in a business suit running to catch a bus, you might have assumed that she was worried about being late for work. If you saw a young man pacing in front of a movie theater, you might have assumed that he was concerned because his date was late. If you saw a small child throwing a temper tantrum, you might have assumed that he was tired or hungry. In each of these cases, you were drawing conclusions about people's internal states based upon the behavior that you observed. Often, internal states—like being concerned, worried, tired, or hungry—influence the ways that people act. All creatures and objects, even subatomic particles, are affected by their internal states. Modeling the internal states of individuals provides you with an opportunity to understand a great variety of real-world systems.

When introducing new variables (or states) for the turtles in your StarLogo models, it is often helpful to envision an animal or object that you are trying to describe. Perhaps you are thinking about simulating a real turtle walking around. Such a turtle might have variables like speed, stamina, age, size, and so on. Once you have identified these variables you have arrived at the first difficult moment—deciding which of these are the most important for what you are modeling. When constructing models in StarLogo, as in many modeling programs, less is more. It is better to determine which variables are most useful and interesting in the context of your model than it is to try to incorporate too many variables. Adding too many variables can make it difficult to interpret or analyze your model. It is usually a good idea to start by picking one turtle variable with which to work. In this example, speed might be particularly relevant, so you could implement that variable.

MODELING CONCEPTS

- Explore ways that individuals react to their internal states.
- Investigate how altering creatures' internal states leads to modifications in their behavior.
- Experiment with methods for deducing individuals' internal states based on their observable behaviors.
- Relate the use of turtle variables and states to variables and states in real-world systems.

STARLOGO CONCEPTS

- Learn about existing turtle variables and how to use them.
- Create and modify your own turtle variables.
- Identify patch and global variables.

Keep in mind that turtle actions can be described at two different levels. On the top level, you can describe a turtle's actions in a model by watching what it does when the simulation is running. On the bottom level, you can look at the rules and states that determine the turtle's actions by reading the project code or looking inside the turtle to examine its states. You can think about turtle behaviors from the perspective of either of these two levels. Since turtle behaviors are based on rules and variables that you create, you can think about building those behaviors from the "bottom up." Or you can spend time observing the turtles' behavior from the "top down." As you watch the turtles, you might be able to make assumptions about their underlying rules or states based on your observations. This process of concurrently building behaviors from the bottom up and investigating behaviors from the top down is a great way to start thinking more deeply about the models that you are building in StarLogo.

CHALLENGE DESCRIPTION

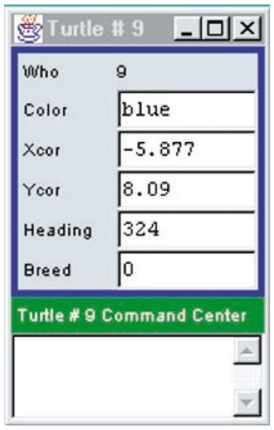
By now you have played with most of the basic turtle variables including turtle number (**who**), **heading**, **color**, **xcor**, and **ycor**. Every turtle in each of your projects has all of these properties. With just these few properties you have managed to do some fantastic things with the turtles. Now you will learn how to add your own variables to the turtles, giving you and your turtles even more flexibility.

In this Challenge, you add the variable **energy**. Your task is to think of a creative project that uses this variable. The value of a turtle's **energy** could change when it runs into a special patch in the environment. Alternatively, you could implement spontaneous **energy** changes in your project. The changes in a turtle's **energy** should have some perceptible effect on its behavior. You might find that it helps to think of objects that have energy (e.g., bouncing electrons or foraging lizards) when deciding how to use this variable. Or you might choose to give your turtles **energy** first and then see what it can make them do. Both approaches are useful and can be effective.

CHALLENGE GUIDELINES

Turtle Monitors

The technique of adding turtle variables is not difficult, but using them effectively can be a bit of a challenge (no pun intended!). This concept may be easier to understand if you first think about turtle variables that already exist. Create some turtles and double-click on a turtle to bring up its **Turtle Monitor**.



Who	9
Color	blue
Xcor	-5.877
Ycor	8.09
Heading	324
Breed	0

A Turtle Monitor.

The **Turtle Monitor** lists and tracks the values of all of the turtle variables. You will notice that the turtle already has certain variables like **color**, **xcor** and **ycor** (the x- and y-coordinates respectively), **heading**, and so forth that enable the turtle to keep track of things like what it looks like and where it is going. Click on the value for **color**, type in a new value (or name) for **color** inside the **Turtle Monitor**, and then press **Return** to see the effect of your change [15]. What happened to your turtle?

Each turtle knows the values of its own variables and can ask other turtles about the values of their variables (you will see this in a later Challenge). In addition to the predefined turtle variables, you can add your own variables if you want your turtles to keep track of additional information.

Creating Turtle Variables

You create turtle variables by typing **turtles-own [*variablenames*]** at the top of the **Turtle Procedures Pane**. ***variablenames*** are the one-word names used to identify the new turtle variables and should be separated by spaces. For example, if you would like to create turtle variables for **energy**, **age**, and **gender**, you would use the statement:

```
turtles-own [energy age gender]
```

In this example, the variables **energy**, **age**, and **gender** are created for every turtle. Notice that there are spaces, not commas, between the variables and that you must list the variables in between square brackets. Make sure that you put all of your turtle variables on a single line. The following code will **not** work correctly:

```
turtles-own [energy]
turtles-own [age]
turtles-own [gender]
```

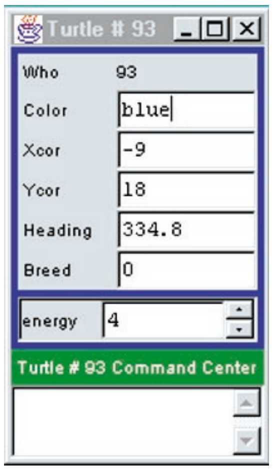
turtles-own should be typed only once in any project.

Monitoring Turtle Variables

Try creating a turtle variable **energy** and setting it to a value (see the hints below). You can check the value of this variable for any turtle by opening its **Turtle Monitor**. Here is a **Turtle Monitor** showing values for **energy** as well as the turtles' standard variables (**color**, **heading**, and so forth.).

Some of the tasks you might want to accomplish with your new variables can be implemented using the following commands. The commands on the next page are based on the sample turtle variable **energy** and can only be executed by the turtles.

A Turtle Monitor with an energy variable.



Turtle # 93

Who	93
Color	blue
Xcor	-9
Ycor	18
Heading	334.8
Breed	0
energy	4

Turtle # 93 Command Center

CHALLENGE HINTS

If you want to...	Use these commands:	For this character:
Create a new variable energy for all turtles	The statement turtles-own [energy] creates a new variable energy and allows you to use the set of commands that manipulate the turtles' energy.	🐢 This statement should be placed at the top of the **Turtle Procedures Pane**.
Set the value of energy for all turtles	The command setenergy *value* sets the variable energy to the *value* specified. Note that setenergy is one word.	🐢
Increase the value of energy for all turtles	The command setenergy energy + *increase* adds the amount *increase* to the current energy level. Note that this is really the same as the last entry (setenergy *value*) with the *value* equal to the current energy plus an *increase*. Also, note the spaces on both sides of the plus sign.	🐢
Do something to a turtle if its energy is above some value	Use the command if energy > *check* [*statements*] to cause turtles with an energy value greater than *check* to perform the *statements*.	🐢
Multiply the energy of turtles who satisfy a certain condition	Use the statement if *condition* [setenergy energy * *multiplier*] to ask those turtles who satisfy the *condition* to multiply their energy by the *multiplier*. Try: if color = blue [setenergy energy * 1.5].	🐢

While these hints just show commands for the sample variable energy, you can use them for any variable that you create. For instance, if you created a variable called age you could use the command setage. For a complete list of the commands that are created for a new variable, check out the StarLogo Documentation section on variables at http://www.media.mit.edu/starlogo/documentation/variables.htm.

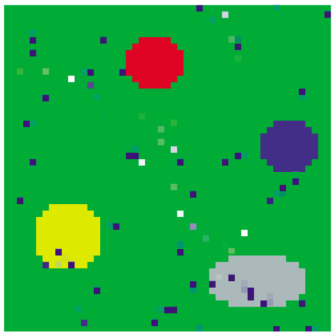
Patch and Global Variables

In addition to turtle variables, there are two other types of variables in StarLogo. Patch variables provide patches with the same ability to store information as turtle variables provide for turtles. If you are interested in implementing patch variables, you can read about them in the StarLogo Documentation.

The observer controls global variables that are not associated with patches or turtles. While every turtle or every patch keeps track of values for each turtle or patch variable, only one value exists for a global variable at any given time. An example of a global variable that you could create is **time**, as the value of **time** is usually the same for every object in the model. Integrating multiple variable types in a single project is a complicated endeavor. We recommend starting with turtle variables, which are the focus of this Challenge.

Energizer Turtles

A straightforward example of using a variable is shown in the project ***Energizer Turtles***. This project builds on the ***Bumper Turtles*** series of projects. In this case when the turtles walk over patches of different colors their **energy** either increases or decreases, depending on the color of the patch that they pass. The turtles gain **energy** if they step on a red or yellow patch. They lose **energy** if they step on a blue or gray patch. The energy level of each turtle is indicated by its color (a more energetic turtle is brighter) as well as by how fast it moves (a more energetic turtle moves faster).

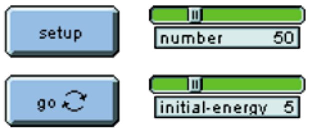

Energizer Turtles' Graphics Canvas.

As you explore this project, there are a couple of details to notice in the procedures. First, a modified **check-patches** procedure asks each turtle to increase or decrease its **energy** depending on the patch color at its location. Second, a turtle's **energy** affects both its color and its speed. The **fd** command uses the value of the turtle's **energy** variable to determine the size of the turtle's forward step each time it moves. The **scale-color** command sets the turtles with the most **energy** to almost white and the least **energy** to almost black, with the turtles in between showing varying shades of **blue**. The **scale-color** command takes four arguments (or inputs), so it is worth a quick look. This program uses the following command:

```
scale-color blue energy 0 20
```

This command literally means, scale the color of the turtles to a shade of **blue**, depending on the value of the **energy** variable, with an **energy** value of **0** resulting in very dark blue and **20** in very light blue.

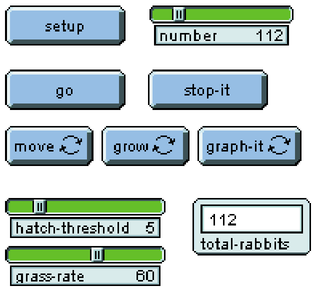
To try this project, click the **setup** button to create the number of turtles shown on the **number** slider. You can add "energy-change patches" by drawing red, yellow, blue, or gray patches. Click the **go** button to start the simulation and see the effect of the colored patches on the color and the speed of the turtles. You may notice that turtles tend to get stuck in some of the patches that decrease their **energy**. Can you modify the percentage increase or decrease in the turtles' **energy** (in the Turtle Procedures) to change this tendency?



Energizer Turtles' Interface.

Rabbits

Another project that uses the energy concept is ***Rabbits***. This project explores a simple ecosystem made up of rabbits and grass. The rabbits wander around randomly, and the grass grows randomly. Rabbits use up **energy** as they move. When a rabbit bumps into some grass, it eats the grass and gains **energy**. If the rabbit gains enough **energy**, then it "reproduces" by hatching a new rabbit. If it loses all of its **energy**, then the rabbit dies.

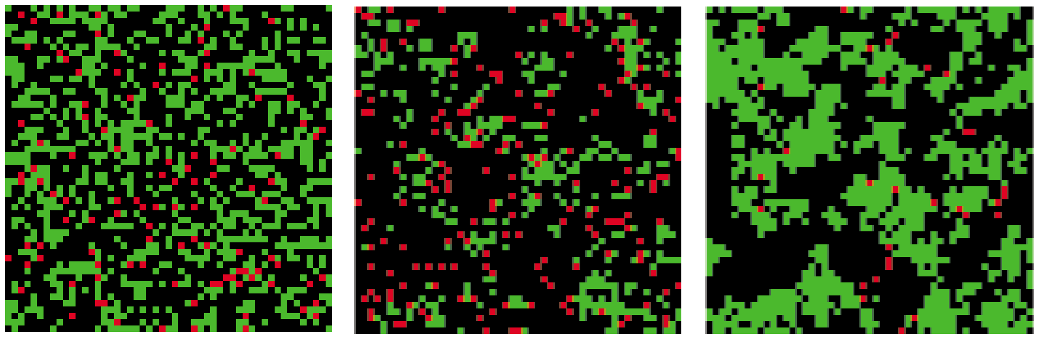


Rabbits' Interface.

Click the **setup** button to set up the rabbits (red) and grass (green). The **number** slider controls the initial number of rabbits. Click the **go** button to start the simulation. The **hatch-threshold** slider sets the energy level at which the rabbits reproduce. The **grass-rate** slider controls the rate at which the grass grows. (Note: You cannot change the **grass-rate** in the middle of a simulation. The change takes effect at the next **setup**.)

Watch the **total-rabbits** monitor to see how the rabbit population changes over time. At first, there is not enough grass for the rabbits, and many rabbits die. But this allows the grass to grow more freely, providing an abundance of food for the remaining rabbits. These rabbits gain **energy** and reproduce. The abundance of rabbits leads to a shortage of grass, and the cycle begins again. Lotka (1925/1956) and Volterra (1926) originally characterized these patterns using a set of differential equations. If you would like to see a graph of these patterns, you can look at the **Plot Window** (in Challenge 8 you will learn to create your own graphs). In order to fit the number of rabbits and the amount of grass on the same scale, the plot shows the number of rabbits and one-fifth of the amount of grass. What happens to the number of rabbits and amount of grass over time? What does this imply for the long-term stability of this population of rabbits and grass? Try manipulating the parameters to change these oscillations.

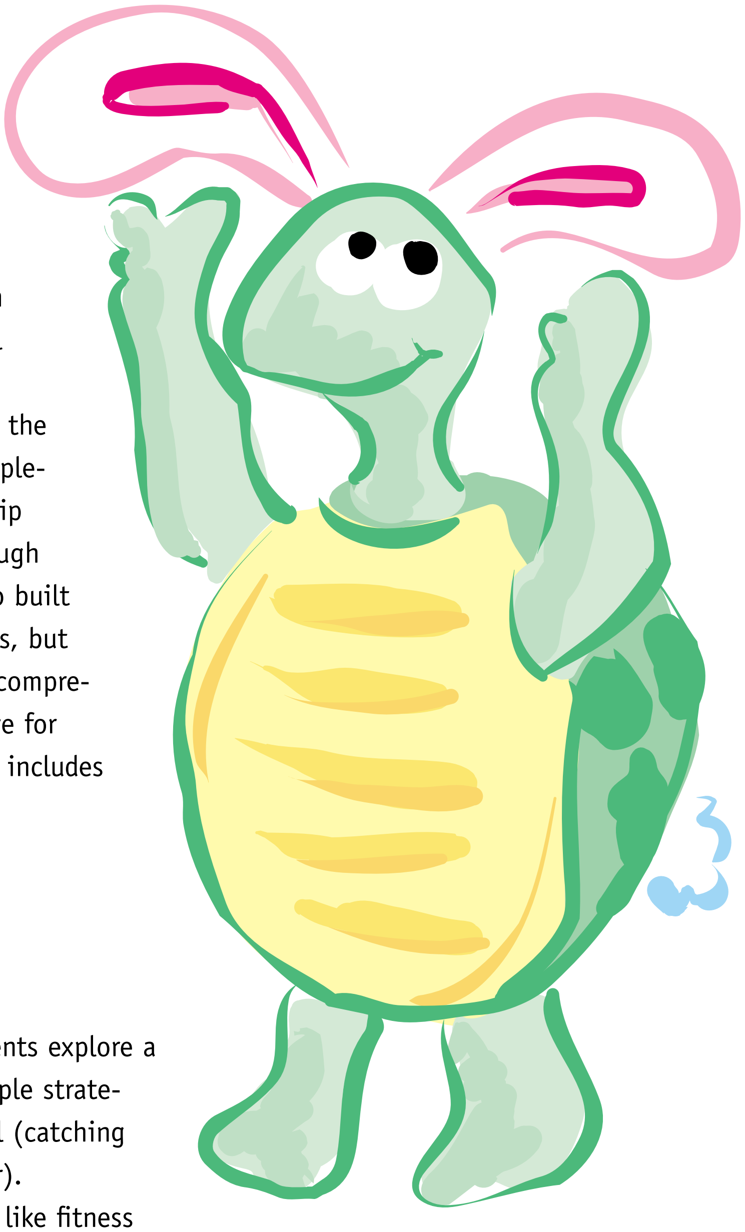
In this model, a single rabbit is able to reproduce, hatching new rabbits whenever its energy level exceeds the threshold. Obviously, this process is an oversimplification of the way that real rabbits reproduce. Some people find it difficult to accept this simplified model of rabbit reproduction. The sample project, ***Rabbits***, is an idea model (see Chapter 2) of rabbits and grass interacting in an ecosystem. It does not attempt to recreate all of the complexity of a real ecosystem. Instead it focuses on just a few relationships (between the rabbits and their food). One way to conceptu-



Rabbits' Graphics Canvas over time.

alize these simplified rabbits is to think of them as "wrabbits," where the silent "w" differentiates between a real rabbit and an abstract, simplified version of a rabbit (a wrabbit) that embodies just a few characteristics of a real rabbit.

No matter how hard you try, you will not be able to model everything about a real rabbit (or any organism) in your projects. Instead, it is best to focus on the essential parts of your model, by thinking about what you want to investigate or explore. In this case, the model focuses on the way that **energy** is incorporated into the system. This implementation of wrabbits succeeds in showing the relationship between food depletion and population growth, even though the model of reproduction is not realistic. The person who built this model *could* have included other rabbit characteristics, but sometimes, additional complexity can make a model less comprehensible. Remember that idea models can be very effective for exploring the dynamics of a particular concept. Chapter 2 includes a detailed discussion of this and other pertinent modeling issues.



TIPS FOR TEACHERS

In Activity 5, Survival of the Fittest Paper Catchers, students explore a model of population growth. That model incorporates simple strategies for implementing reproduction and fitness in a model (catching the piece of paper while keeping a foot on the newspaper). Students might consider ways to model similar processes, like fitness or reproduction, in a StarLogo project. Perhaps when their turtles reach a certain energy level, they are able to hatch new turtles. Of course, students will want to think about the states of the newly hatched turtles (i.e., How much **energy** do they have? What is their age?). They might also explore what kind of population growth occurs in that model. Regulating the total amount of available **energy** in the environment (like regulating the size of the paper or the ability of the paper catchers) can have a dramatic impact on the rate of population growth.

During their work-in-progress reports, students should be able to articulate how and why the turtles' energy levels change and describe the effects that those changes have on the turtles' behavior. They should also be able to describe how one could hypothesize about the level of **energy** that a turtle possesses by observing the turtle's actions. Students should be able to give a rational explanation for the turtle behavior that they created. Some of these explanations might draw parallels to the states of creatures in the real world (like hunger, age, or happiness).