

GridWorld

Activity 5 - Roadrunner 2

Objective: To introduce a “stunned” Coyote and Roadrunner characters into the Roadrunner simulation in GridWorld.

Background:

Another cartoon character, the Roadrunner, is a fast-footed feathered flightless bird. It lives in Wile E Coyote’s world that we created in part 1 of this activity. The Roadrunner likes to run quickly from place to place looking for food. The Coyote tries to catch and eat the Roadrunner by placing exploding Boulders around the grid. Here is what the Roadrunner looks like:



The Roadrunner is so fast, he moves several cells at once. With that speed, he can often run into other things like Stones, Boulders, and Coyotes. Stones and the edge of the grid are not a problem, he just stops before he hits them. Boulders are a problem because the Boulder explodes and the Roadrunner is removed from the grid. When a Roadrunner hits a Coyote, the Roadrunner is fine, but the Coyote is knocked back and stunned for some time.

Your job is to introduce both the “stunned” Coyote and the Roadrunner characters and their behaviors into this world.

Assignment:

1. Create a **SickCoyote** class that extends the **Actor** class. The SickCoyote will act like the stunned Coyote after he is hit by the Roadrunner. He has two fields like a **Boulder**: a lifetime and a threshold length of lifetime. Make the threshold equal to 10. When a **SickCoyote** is created, set the color to null and lifetime to the threshold. A second constructor should pass an integer parameter that sets its lifetime. **SickCoyote's** **act()** counts down its lifetime. When the lifetime reaches zero, a new (healthy) **Coyote** is created and replaces the **SickCoyote**.

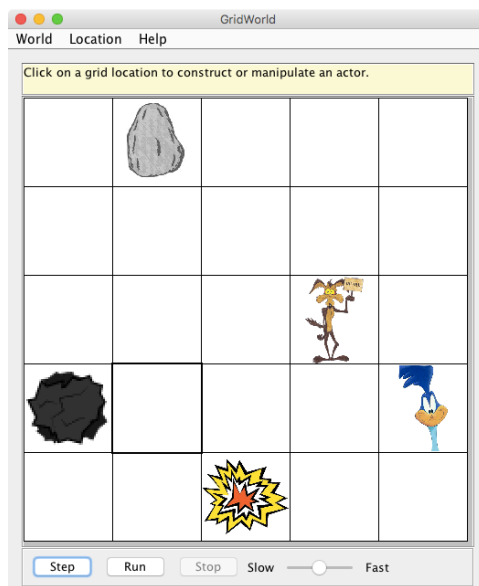
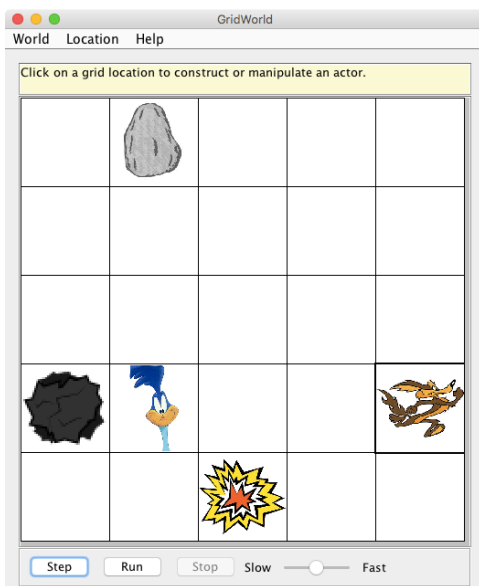


2. Test your **SickCoyote** class. Create a **SickCoyoteRunner** that places 4 **SickCoyotes** with different lifetime parameters on a 5 by 5 grid. Run the simulation and watch to see that all **SickCoyotes** are replaced by **Coyotes**.
3. Create an **RR** class (Roadrunner) that extends the **Critter** class. An **RR** runs in a straight line three cells at a time or when it runs into an actor or the edge of the grid, whichever comes first.

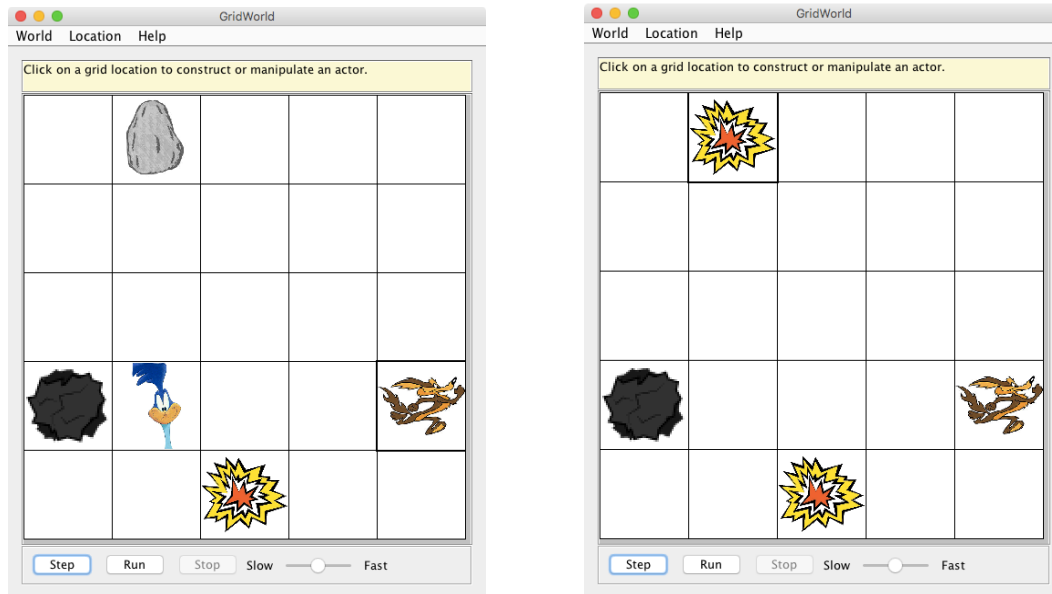
The **RR** works in this way:

- a. When an **RR** is created, its color is null and its direction is north. An **RR** does not process any actors.
- b. The **RR** gets all possible locations to move that form a clear path (empty cells) between him and his destination up to 3 cells away in a compass direction (north, northeast, etc). An **RR** can “bump” into **Boulders** and **Coyotes**, which means an **RR** can land on their locations. An **RR** never “bumps” into **Stones**, **SickCoyotes**, or **Kabooms** and always chooses empty locations short of running into those actors or off the grid.
- c. **RR** selects its destination location at random from its list of possible locations.
- d. When an **RR** lands on a **Boulder**, the **Boulder** explodes and **RR** goes bye-bye. In effect, the **Boulder** is replaced with a **Kaboom** and the **RR** is removed from the grid.
- e. When an **RR** lands on a **Coyote**, the **RR** “hurts” the **Coyote** and bumps him off his cell. This means the **Coyote** is removed from the grid, the **RR** is moved into the **Coyote’s** location, and a new **SickCoyote** is placed in a random adjacent empty cell. (The **SickCoyote** will eventually replace itself with a “well” **Coyote**.)

Here are some examples of “bumping”. If **RR** “bumps” into the **Coyote**, he takes the **Coyote’s** spot and a new **SickCoyote** is placed in an adjacent empty cell (figure on the right).



If **RR** “bumps” into the **Boulder**, **RR** is removed from the grid and the **Boulder** is replaced with a **Kaboom** (figure on the right).



6. Mr Greenstein provided a **RRRunner** class in your zip file that places **Coyotes** and **RRs** on a 10 by 10 grid. Use **RRRunner** to test your code.