CompSci 4
Chap 4 Sec 1
Sept 13, 2007

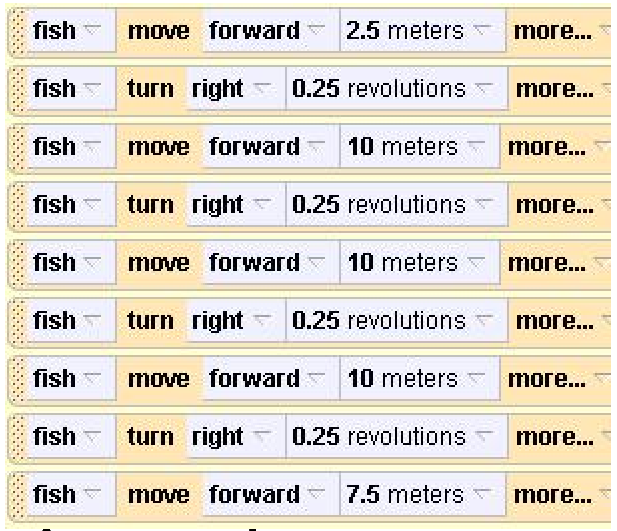
Prof. Susan Rodger

Announcements

- Read Chapter 4, Section 2 for next time
- Assignment 3 storyboard due Tuesday
  - World is due next Thursday

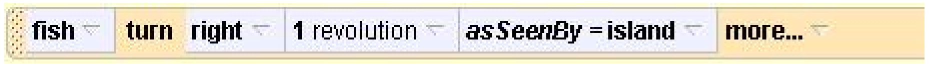
Review

- Fish circling around island



```
fish move forward 2.5 meters more...
fish turn right 0.25 revolutions more...
fish move forward 10 meters more...
fish turn right 0.25 revolutions more...
fish move forward 10 meters more...
fish turn right 0.25 revolutions more...
fish move forward 10 meters more...
fish turn right 0.25 revolutions more...
fish move forward 7.5 meters more...
```

jagged

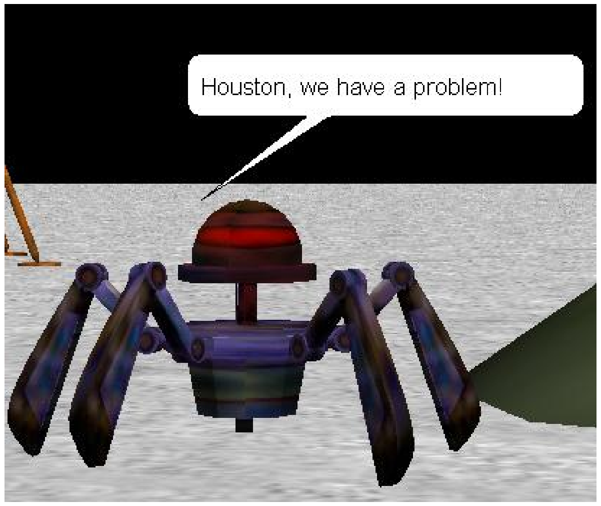


```
fish turn right 1 revolution asSeenBy = island more...
```

smooth

Show world

What we will do today



- Lecture on Chap 4, Sec 1
- Classwork
  - Create three animations
    - Snowpeople mods including flipping hats
    - Helicopter
    - Cameras moving
  - Get checked off today and for last time

Larger Programs

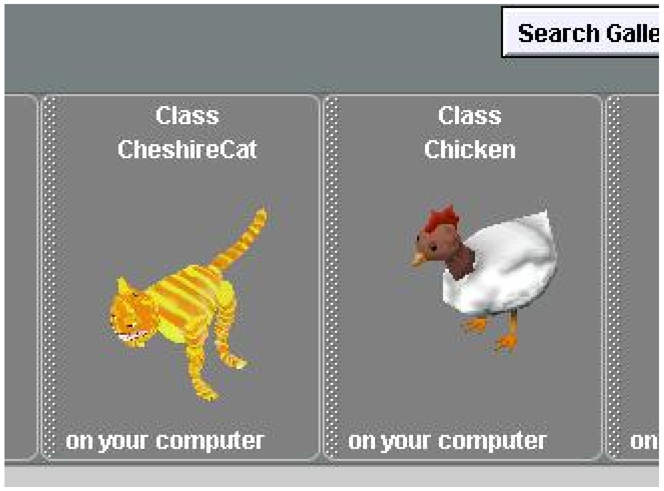
- Programs start to increase in size – many lines of code
- Games and “real world” applications have thousands, even millions of lines of code
- Want to organize large programs into small manageable pieces

Classes, Objects and Methods

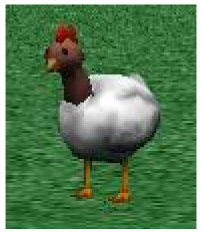
- Object-oriented programming uses classes, objects and methods as basic components
- These components help you
  - Organize large program into small pieces
  - Design and think about an intricate program
  - Find and remove errors (bugs)

In your programs, you've used

- Classes
  - In Alice, classes are predefined as 3D models



- Objects
  - An object is an instance of a class
    - Class: Chicken
    - Objects: Chicken, Chicken2



In your programs, you’ve also used

- Built-in (predefined) methods
  - Examples: move, turn to face, say

- World.my first method
  - Example: robot on the moon from chapter 2, wrote code where an alien surprised the robot
  - All the code in World.my first method

Modifying the Program

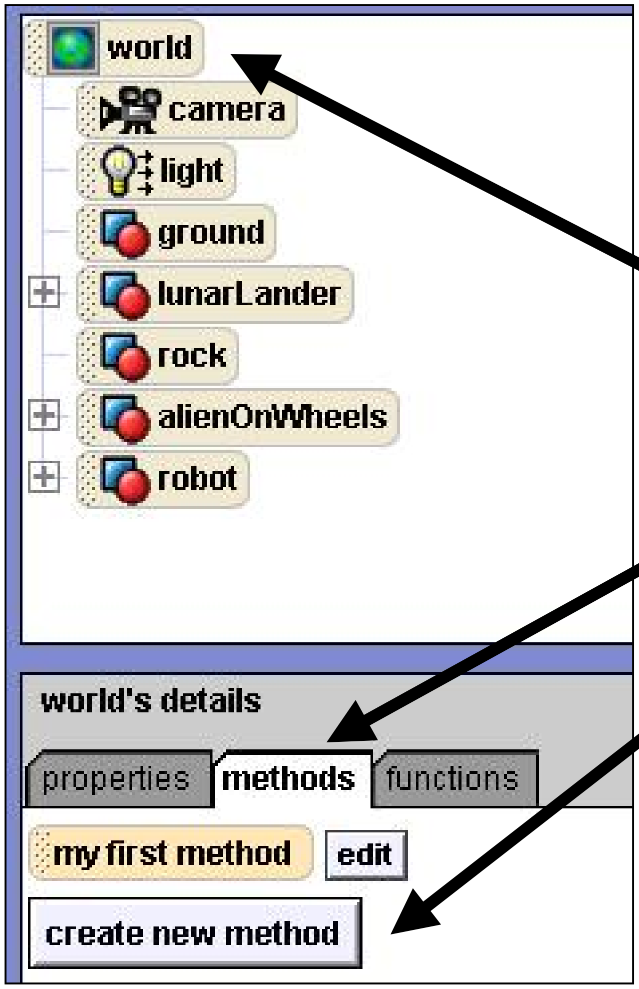
- Modify program to get robot to try twice to move toward the alien or the alien go up and down twice.
- To make modification, add more lines of code
  - makes the program code longer and more difficult to read and think about
- Show alien world from last time

A Solution

- A solution to the problem is to
  - Define our own method
  - Name the new method `surprise`

- Then, can drag-and-drop the `surprise` method into the edit box, just like the built-in methods

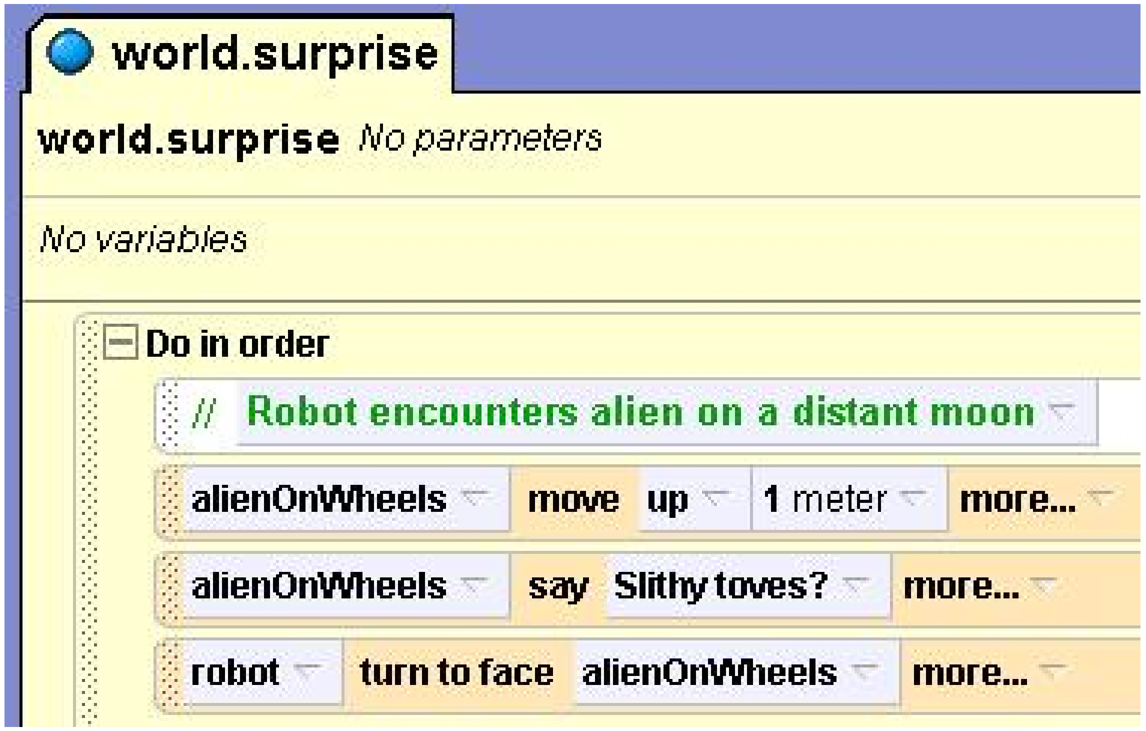
Demo: The Solution



- First associate new method with the world
- Select World tile
- Select methods tab
- Click on “create new method”
- Demo

World-level method

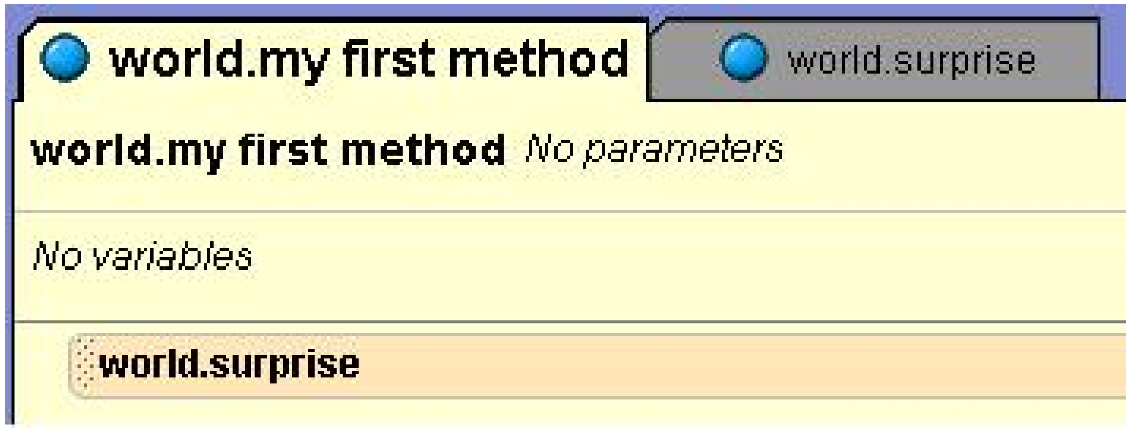
- surprise is a world-level method because it
  - Is defined as a method for World
  - Has instructions that involve more than one object (robot,alienOnWheels)



```
world.surprise
world.surprise  No parameters
No variables
Do in order
    // Robot encounters alien on a distant moon
    alienOnWheels  move  up  1 meter  more...
    alienOnWheels  say  Slithy toves?  more...
    robot  turn to face  alienOnWheels  more...
```

Using the surprise method

- This method is executed by calling (invoking) the method from my first method

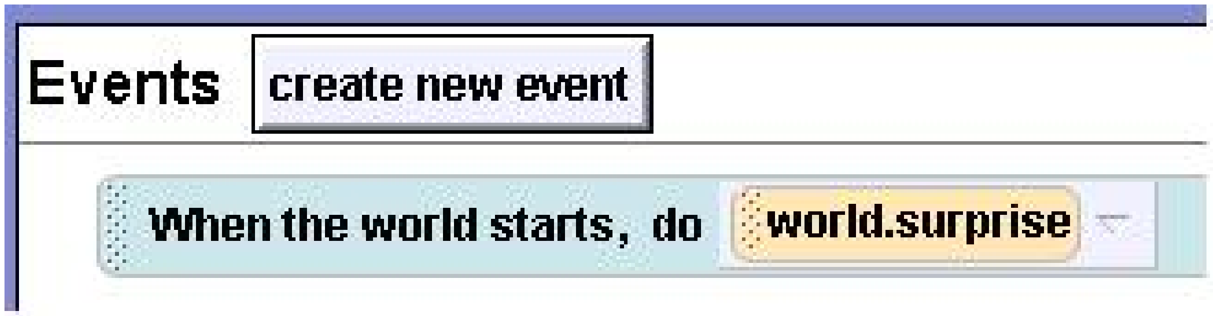


world.my first method *No parameters*

No variables

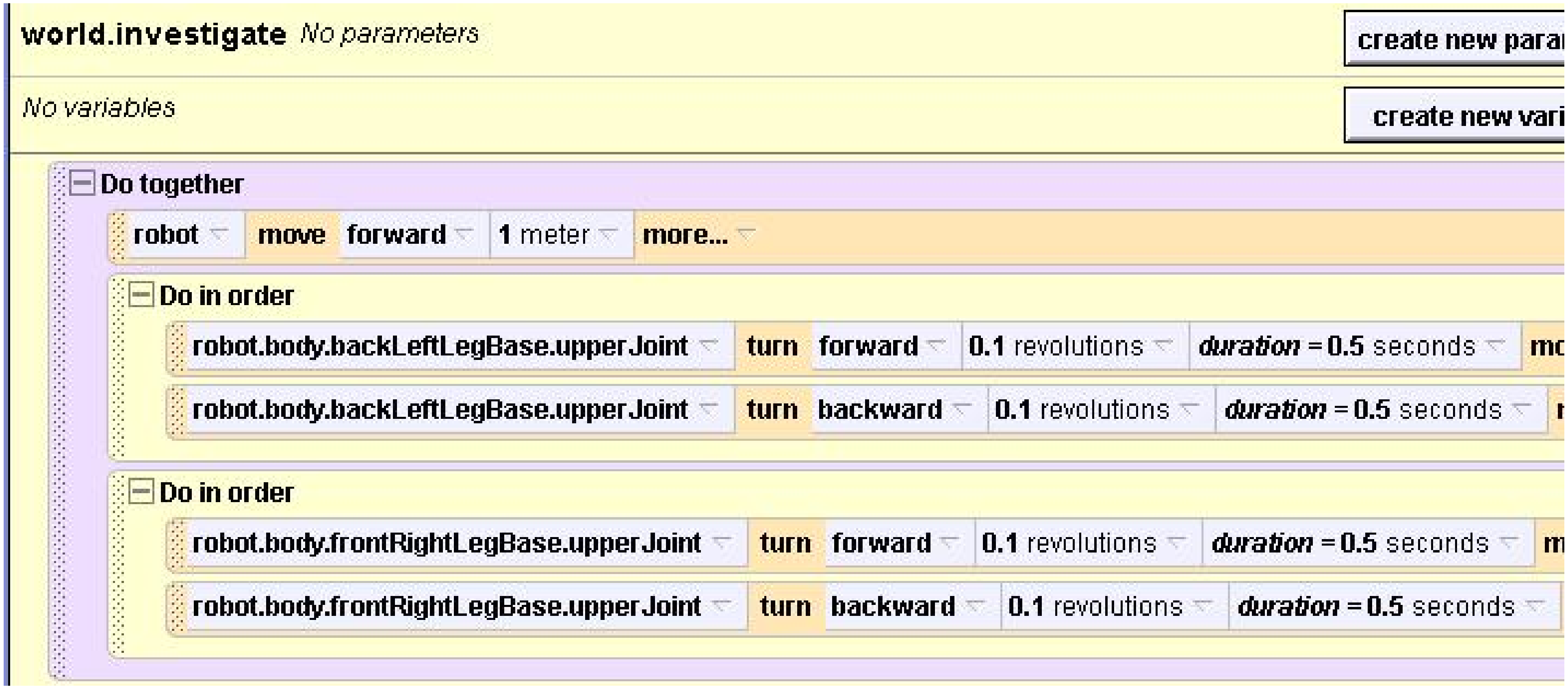
world.surprise

- For testing, invoke temporarily when world starts



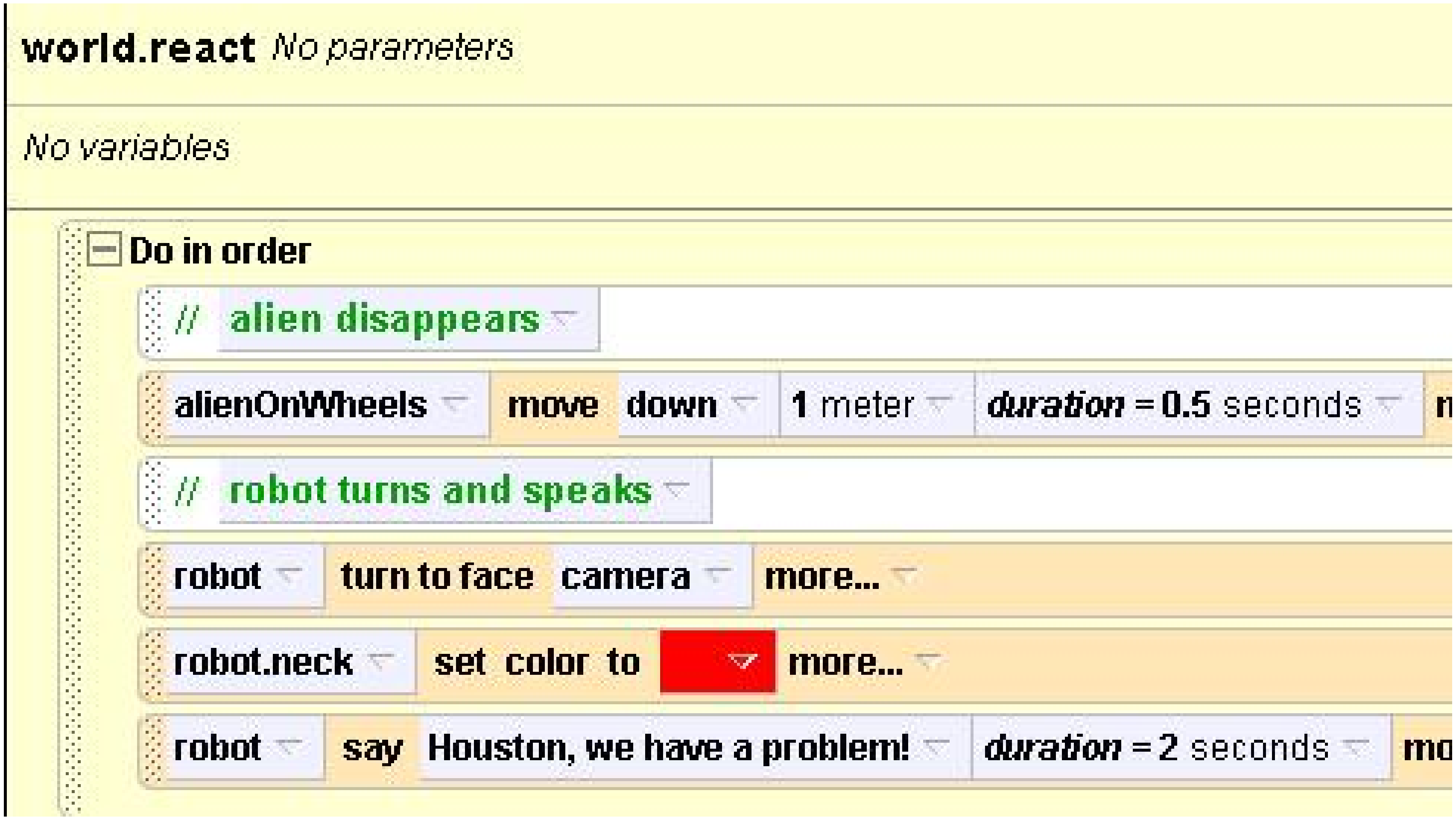
When the world starts, do world.surprise

investigate method



```
world.investigate No parameters
No variables
Do together
    robot move forward 1 meter more...
    Do in order
        robot.body.backLeftLegBase.upperJoint turn forward 0.1 revolutions duration = 0.5 seconds mo
        robot.body.backLeftLegBase.upperJoint turn backward 0.1 revolutions duration = 0.5 seconds r
    Do in order
        robot.body.frontRightLegBase.upperJoint turn forward 0.1 revolutions duration = 0.5 seconds m
        robot.body.frontRightLegBase.upperJoint turn backward 0.1 revolutions duration = 0.5 seconds
```

react method



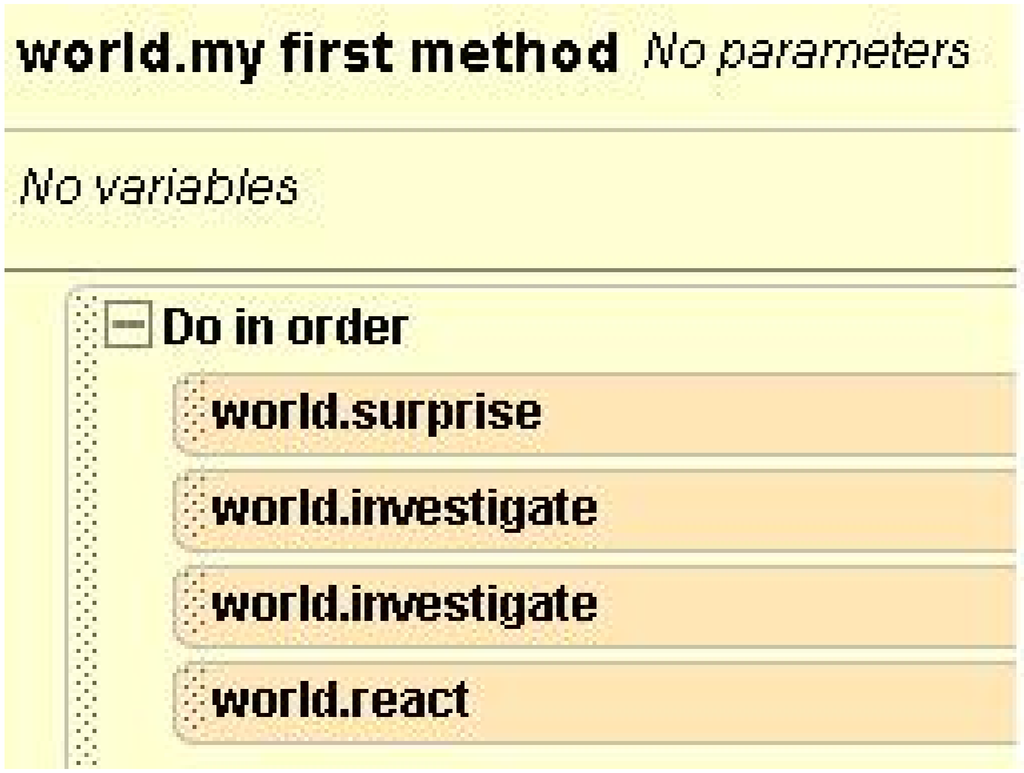
```
world.react No parameters

No variables

Do in order
    // alien disappears
    alienOnWheels move down 1 meter duration = 0.5 seconds m
    // robot turns and speaks
    robot turn to face camera more...
    robot.neck set color to 🟥 more...
    robot say Houston, we have a problem! duration = 2 seconds mo
```

Why write our own Methods?

- Saves time – can call method again and again without rewriting code
- Reduces code size – call method instead of rewriting same code
- Allows us to think at higher level
  - Think “surprise” instead of “alien moves up, alien says something, robot turns around…”
  - Technical term for “think at a higher level” is abstraction

World.myFirstMethod now

- Move robot forward twice as far by invoking “investigate” twice



```
world.my first method  No parameters

No variables

  Do in order
      world.surprise
      world.investigate
      world.investigate
      world.react
```

Classwork today

- Modify snowpeople to add two methods
  - catchAttention
  - Fliphats
- Move the camera with an object
  - skyride – download from CompSci 4 page
- Create airport/helicopter world with new method
  - circleTower