CompSci 94
Flexible Procedures
February 6, 2018

Prof. Susan Rodger
Class Today

• Making procedures more flexible – adding in parameters
• Loops - repetition

• Assignment 3 due Thursday night, Feb 8
  – Turn in Alice worlds using websubmit
  – Build and link to two web pages
  – Put storyboards on web pages
Review 1

• Where in Alice do you add a cameraMarker?

• In my code, how do I place the camera on the cameraMarker so it is looking the direction of the cameraMarker?
Review 1

• Where in Alice do you add a cameraMarker?
  – Click on “setup scene”, on the right scroll down and click on “add cameraMarker”

• In my code, how do I place the camera on the cameraMarker so it is looking the direction of the cameraMarker?
  – Select “Camera” and drag in the procedure moveAndOrientTo and select the cameraMarker
Review 2

• How do I get a Hare to circle halfway around a cow in a forward direction if they are both facing front and the Hare is to the right of the cow?
Review 2

• How do I get a Hare to circle halfway around a cow in a forward direction if they are both facing front and the Hare is to the right of the cow?
Review 3

• If I want to teach the cow to Dance. How would I create the Dance procedure in Alice?

– Click on Scene tab, then Cow, then Add Cow procedure, and give it the name Dance

• How do I test just my Cow Dance procedure?

– Click on “InitializeEventListeners – Disable myFirstMethod – Drag in Cow Dance
Review 3

• If I want to teach the cow to Dance. How would I create the Dance procedure in Alice?
  – Click on Scene tab, then Cow, then Add Cow procedure, and give it the name Dance

• How do I test just my Cow Dance procedure?
  – Click on “InitializeEventListeners
  – Disable myFirstMethod
  – Drag in Cow Dance
Setup for Lecture Demo

- Four characters: (left to right)
  - MadHatter, Hare, Cow, Bison
  - The Bison is back further
Built-in Procedures are flexible

- Move – pick direction and how far

Drag into the editor

Choose values

Result is
Write procedures with parameters

- A **parameter** is a “place holder” for a value

- An **argument** is the value you put in the place holder
Write procedures with parameters

• A **parameter** is a “place holder” for a value
  
  – *Parameters*: direction, amount

• An **argument** is the value you put in the place holder
  
  – *Arguments*: forward, 2.0
Teach the Hare how to jump – write a procedure

• Click on Scene tab, then Hare, then add Hare procedure
But Wait!!!! A Hare is a Biped! Instead – we could write jump for all bipeds….

- Click on Scene tab, then Biped, then add biped procedure
All bipeds have the same structure
Write jump for all bipeds
How Far to jump?

• Add parameter for distance to jump
  – Type: DecimalNumber
  – Name: distance
Use parameter distance as place holder

• Distance

• Drag over numbers you want to enter
Distance is a place holder for a number

• Distance appears in two places in the code
How high to jump?

• Since we added a parameter for jump, where we call jump, we must now select a number for how high to jump
Add another parameter, how fast to jump
  • Parameter speed, add duration with speed
  • Drag speed over the duration numbers!
Arguments – values passed to procedures

• Add a value for speed

• madHatter is a biped and can also jump
Another new concept - Looping

• We move code to my first method and drag up count from the bottom and pick a number
Write another procedure for hare to ride a quadruped animal

- Write this one as a Hare procedure
- Focus on the hare jumping on and riding the cow
- But really want the hare to be able to ride any quadruped (4 legs)
Another way to add a procedure

- Choose **Hare** from menu
- Name procedure **ride**
Add a parameter `transport`

- The type is a quadruped (from the Gallery class)
- Then name it `transport`
Ride procedure with two parameters transport and distance
Add direction and phrase parameters

• Use new parameters
• Note their types!
Call **ride** procedure twice

- Hare rides cow, then bison
- Bison turns different direction and hare says something different
Final code for ride (part 1)

```
declare procedure ride
with parameters: Quadruped, transport, DecimalNumber distance,
, TurnDirection direction, TextString phrase
```

**do in order**
- `this` turnToFace `transport` add detail
- `this` moveAndOrientTo `transport` add detail
- `this` move UP, `distance`, duration 0.25 add detail

**do together**
- `this` getLeftHip `turn` LEFT, 0.125 add detail
- `this` getRightHip `turn` RIGHT, 0.125 add detail
Final code for ride (part2)

```plaintext
this say phrase add detail

this setVehicle transport

transport turn direction 0.25 add detail

transport move FORWARD 4.0 add detail

this move FORWARD 2.0 add detail

do together
	his getLeftHip turn RIGHT 0.125 add detail

this getRightHip turn LEFT 0.125 add detail

this move DOWN distance add detail
```
This lecture covered

• Looping code a specific number of times with count

• Making procedures more flexible
  – Add parameter as a place holder
  – Plug in an argument when you call the procedure

• Make a procedure for a biped and it works for all bipeds