CompSci 94
Classwork: Scene Activation Listener
Game with two levels
November 12, 2024

Prof. Susan Rodger

1) The story A two-level game

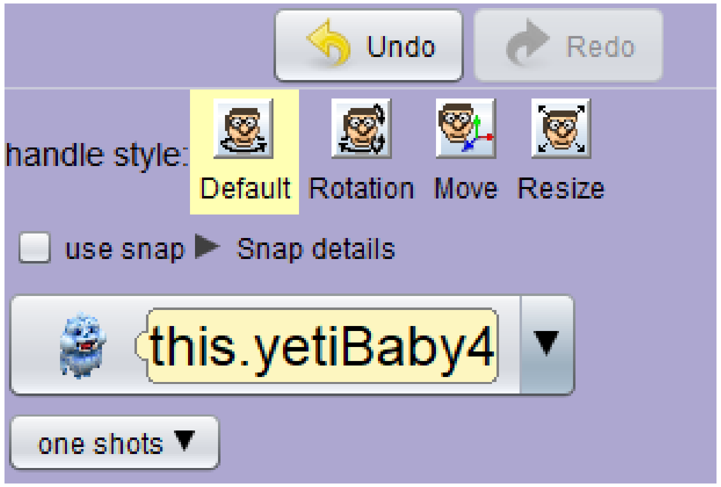
- You will click on the penguin and play the game you made last time, getting the penguin to collide with every yetiBaby. When they are all invisible, then you will click on the penguin again and the yetiBabies will move randomly a different way. You will again collide with all the yetiBabies until the penguin will tell you you have done a great job!
- Follow the steps that follow to build the game.

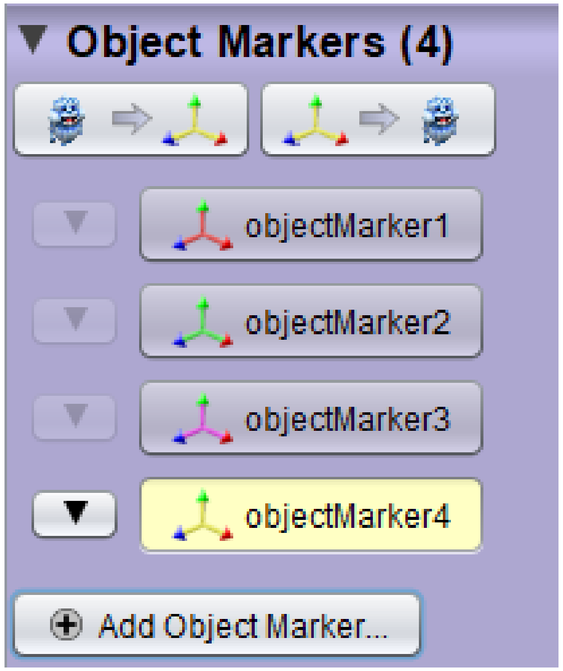
2) Setting up the game

- Start with your Alice world from last time.
- Save the world with FILE SAVE AS and name it something like **classwork19Nov12**
- Create an object marker for each yetiBaby. Be sure to create them **in the same order** the yetiBabies are in the yetiBaby array.
  - Click on yetiBaby, create objectMarker for it
  - Click on yetiBaby2, create objectMarker for it
  - Etc. (see next slide)

2a) Setting up the game

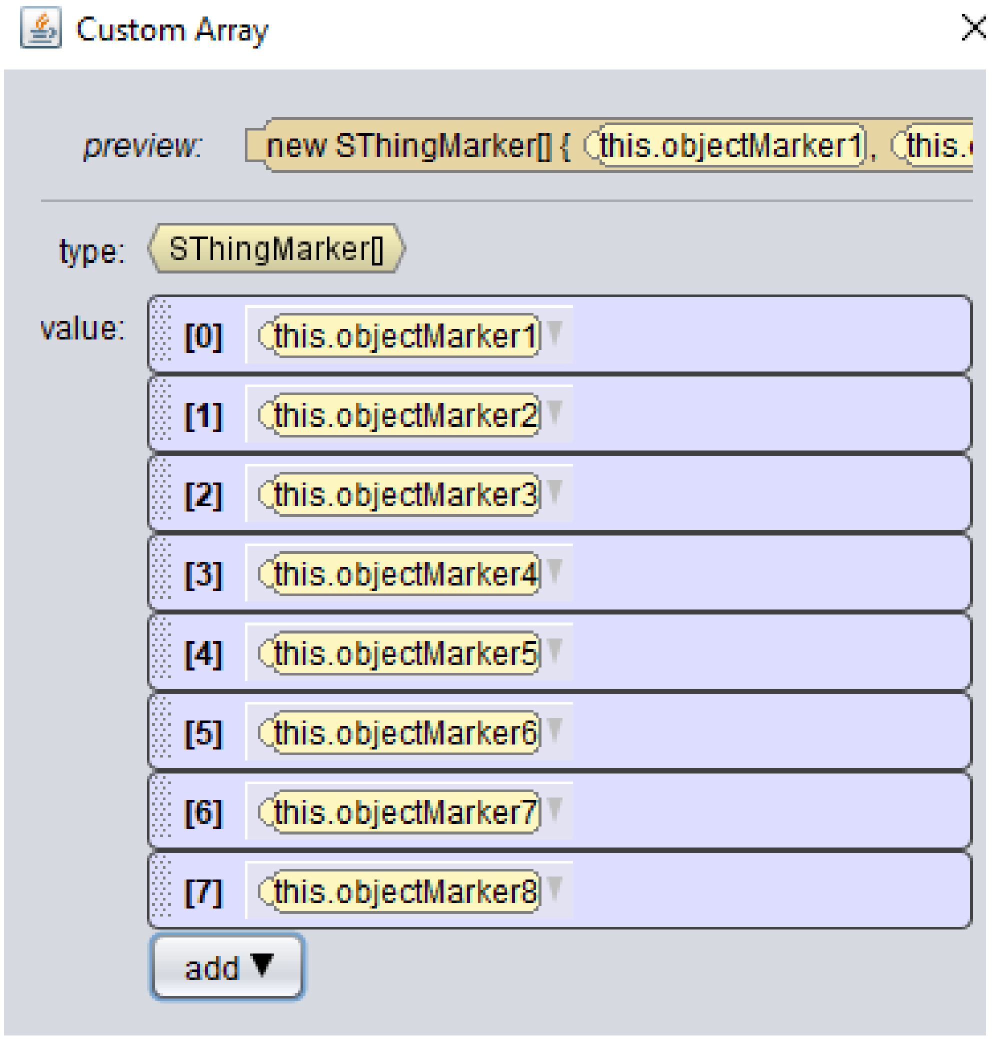
- For example, click on yetiBaby4, create objectMarker4





2b) Setting up game

- Under Scene properties, create an **objectMarker array**, and put all the objectMarker's in it, in order: objectMarker, objectMarker2, objectMarker3, etc.



preview: new SThingMarker[] { this.objectMarker1, this.

type: SThingMarker[]

value:

[0]	this.objectMarker1
[1]	this.objectMarker2
[2]	this.objectMarker3
[3]	this.objectMarker4
[4]	this.objectMarker5
[5]	this.objectMarker6
[6]	this.objectMarker7
[7]	this.objectMarker8

2c) Setting up the game

- Setup an object marker for the penguin
- Create a new **Scene property** of **type TextString** named **gamestate** with initial value **“setuplevelone”**

States for gamestate

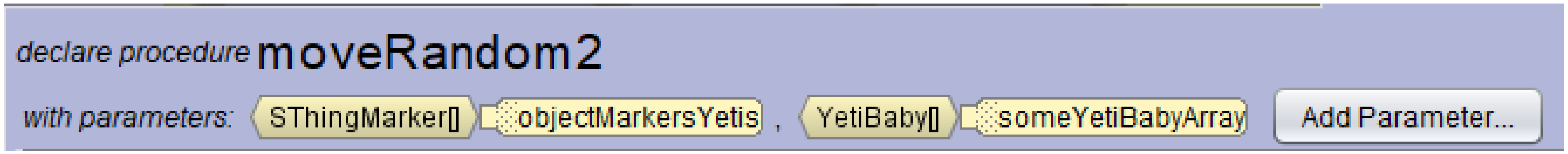
- This is the game flow:
  - Start in “setuplevelone”
  - Change to “levelone” – now play level one game
  - When game one is over, change to “setupleveltwo”
  - Change to “leveltwo” – now play level two game
  - When game two is over, change to “gameover”
- Note: Things for playing the game, such as moving the penguin, or yetiBabies moving, should only work in gamestate levelone or leveltwo

3) Write a **yetiBaby** procedure named **moveRandom2**

- Idea for moving. YetiBaby will move random left/right, move up, move forward/backward, and then move down. If it strays too far away, it will move back to its objectMarker.
- We need access to the objectMarker array and the yetiBabyArray, but we don't have access to them in the yetiBaby procedure. So we will need to send them both as parameters. Details follow….

3a) Write yetiBaby moveRandom2

- Create an **array parameter** of type **SThingMarker** named **objectMarkersYetis**
- Create an array parameter of type **yetiBaby** named **someYetiBabyArray**
- **Make sure you checked the array box for both. You should see the [ ] in the type**



declare procedure moveRandom2
with parameters: SThingMarker[] objectMarkersYetis , YetiBaby[] someYetiBabyArray

3b) Write yetiBaby moveRandom2

- Put in a do in order.
- Create a variable named **leftRight** that is a random number from -2.0 to 2.0
- Create a variable named **backforward** that is a random number from -2.0 to 2.0
- You will need to figure which objectMarker goes with this yetiBaby, so as it moves randomly and strays too far, it can be reset to its objectMarker location. (details next slide)

3c) Write yetiBaby moveRandom2

- You will need an index loop to loop through the yetiBabies, so when **this** is the yetiBaby in the loop, then you know its location in its array, which is the same location for its objectMarker in the objectMarker array.
- see next slide for details on how to do this

3d) Write yetiBaby moveRandom2

- Add this code in this procedure
- Initialize index variable
- Do a count loop through the size of the yetiBaby array
  - If this yetiBaby is the yetiBaby in the yetiBaby array at the current index AND this yetibaby is more than 2 units from its objectMarker
    - Then move the current yetiBaby to its objectMarker
  - Update the index

3e) Write yetiBaby moveRandom2

- Continue with code in the moveRandom2 procedure:
  - Have the yetiBaby move leftright, a random amount, quickly in 0.25 second
  - Next have it move up 1, quickly in 0.25 second
  - Have the yetiBaby move forwardBackward, a random amount, quickly
  - Then have it move down 1.0 in 0.25 seconds
- That's it for this procedure!

4) Rewrite the sceneActivation listener where the yetiBabies move randomly

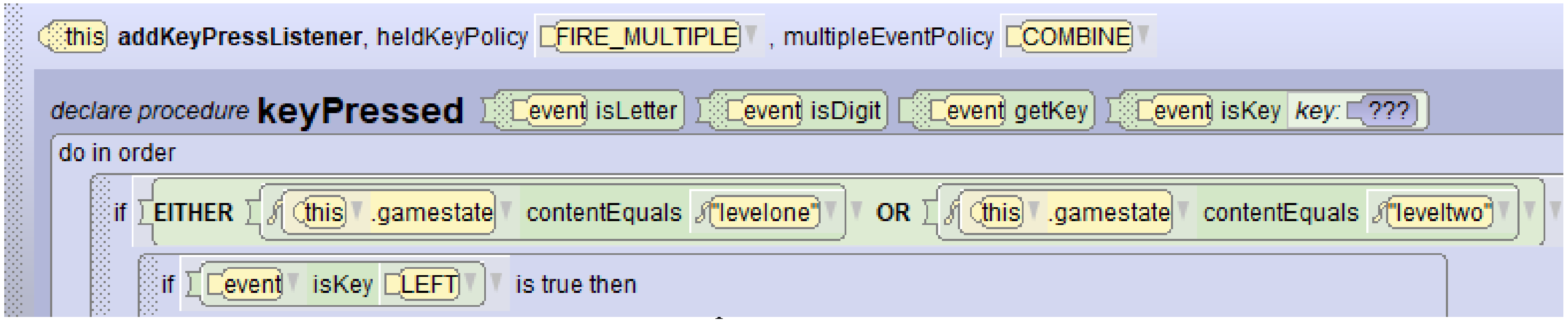
- We will want the yetiBabies to move randomly one way in the first game and then the more complicated way in the second game. If the gameState is “levelone” they will move randomly the simpler way, and if the gameState is “leveltwo” they will move the more complicated way.
- See details on next slide

4a) rewrite sceneActivation listener

- Change the while loop back to **while true**
- If the gameState is “levelone” have all the yetiBabies call moveRandom at the same time
- Else if the gameState is “leveltwo” have all the yetiBabies call moveRandom2 at the same time
- Don’t do anything if the gameState is anything else.

5) Fix the addKeypressListener

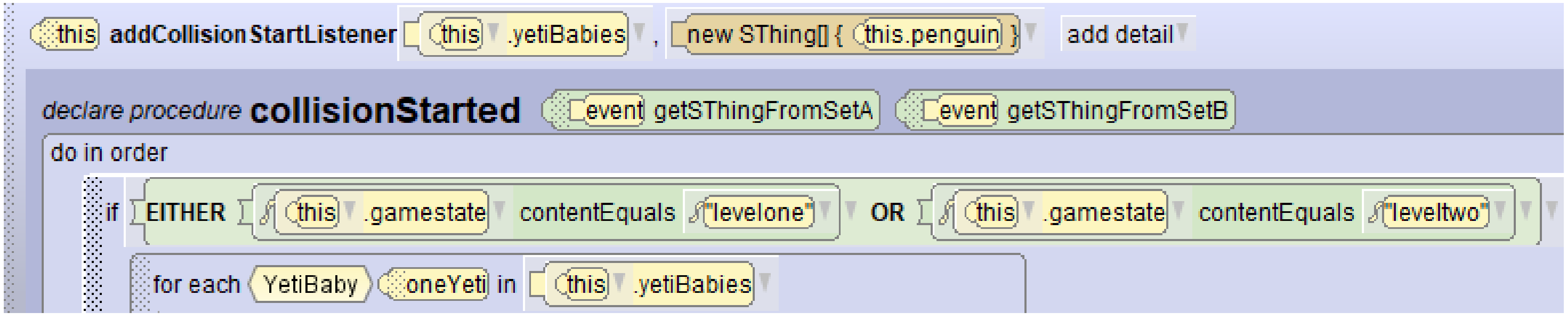
- You only want the penguin to move when you are playing either gameone or gametwo.
- Details:
  - Add an if statement around everything else so you only can move the penguin if the gameState is levelone or leveltwo



```
this addKeyPressListener, heldKeyPolicy FIRE_MULTIPLE , multipleEventPolicy COMBINE
declare procedure keyPressed  event isLetter  event isDigit  event getKey  event isKey key: ???
do in order
    if EITHER this .gamestate contentEquals "levelone" OR this .gamestate contentEquals "leveltwo"
        if event isKey LEFT is true then
```

6) Also guard the collision event

- You only want to make a yeti disappear when there is a collision if you are playing the game, either levelone or leveltwo.
- Details:
  - In the collision event, add an if statement around everything so it is only done if the gameState is levelone or leveltwo



```
this addCollisionStartListener this .yetiBabies , new SThing[] { this.penguin } add detail
declare procedure collisionStarted event getSThingFromSetA event getSThingFromSetB
do in order
    if EITHER this .gamestate contentEquals "levelone" OR this .gamestate contentEquals "leveltwo"
        for each YetiBaby oneYeti in this .yetiBabies
```

7) Add one new event

- You will click on the penguin to start each game.
- Details:
  - Create a new event, a mouseClickOnObjectListener for when you click on a penguin. In this event you will:
    - If the gameState is setuplevelone, then change the gameState to levelone (that means game one will start!)
    - If the gameState is setupleveltwo, then change the gameState to leveltwo (that means the game two will start!)

8) Now it is time to fix myFirstMethod where the game control will be

- MyFirstMethod should do the following:
  - The penguin should say welcome, tell you to steer the penguin into the yetiBabies, and tell you to click on it to start the game.
  - Next have the while loop say: **while isAnyYetiVisible is true**
    - There should be NOTHING in this loop
    - This loop just waits here while you play the first game. This loop will stop automatically when all the yetiBabies are invisible.
  - Go to next slide

8a) myFirstMethod continued

- Immediately after the while loop, change the gameState to “setupleveltwo” (this should make the yetis stop moving)
- Then have the penguin turn to face the front and say “got them all”
- Delete any other code you have in myFirstMethod
- Run your program. You should be able to play the first game.

9) Write the **Scene** procedure named **setupLevel2**

- You will put in here all the setup needed to setup the game to play the second game.
- Add in a do in order.
- You need to loop through all the yetiBabies and have each one move and orient to its objectMarker. You will need an index array loop to do this. (details next slide)

9a) setupLevel2 procedure (cont)

- Create an index variable (whole number) set to 0
- Loop over all the yetiBabies.
  - Have the yetiBaby move and orient to its objectMarker (that is in the index position in the objectMarker array)
  - Update the index variable by 1
- Next have the penguin move to its objectMarker
- Finally make all the yetiBabies visible at the same time.
- That's it for this procedure!

10) Finish up myFirstMethod

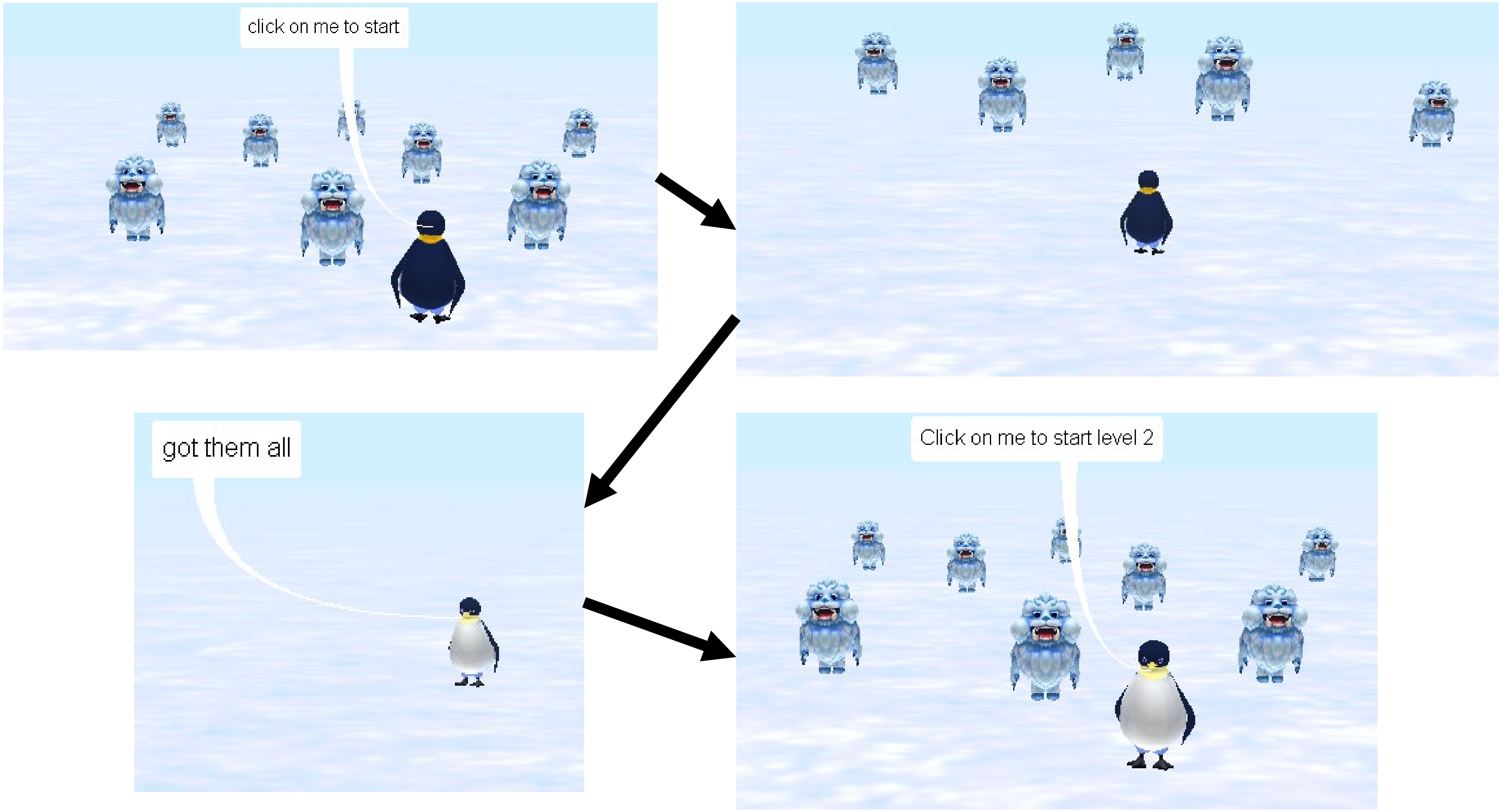
- Add code to the end of myFirstMethod
  - Call setupLevel2
  - Next have the penguin face front
  - Then have the penguin say “click on me to start level 2”
  - Have the penguin turn back around to face the yetiBabies
  - Create another while loop that is: **while isAnyYetiVisible is true**
    - There is NOTHING inside the while loop, it just waits while you play game 2

10a) Finish myFirstMethod (cont)

- Immediately after the while loop, change the gameState to “gameover” (that will stop the yetiBabies from moving)
- Have the penguin turn to face front and say “You did great!”

- That’s it, play your game!

Snapshots



More Snapshots

You did great