

# CompSci 94

## A game with two levels

November 12, 2024



Prof. Susan Rodger

# Announcements

- Assignment 6 (Final project) is out – see deadlines on Assignment page
- TODAY, Nov 12: deadline to request an assigned partner
- Added 4<sup>th</sup> free extension!
- Exam 3 is Nov 19
  - See exam 3 study materials on Nov 19 date on our calendar page
  - Some review today, some next time

# Exam 3 Logistics

- Exam 3 is on Tuesday, Nov 19
- Covers topics through Thursday, Nov 14 lecture
- Old tests are on the calendar web page
- Exam 3 is on paper
- See Exam 3 reference sheet – part of the exam
- Exam 3 is your own work
- Bring only a pen or pencil

# Exam 3 topics

- Topics from last time (loops, arrays, ifs, procedures, parameters, etc)
- Array index loops (see penguin classwork)
- Writing functions
- Event programming
- Changing Scenes (we do on Thursday)

# Events 1

**this** addSceneActivationListener

declare procedure **sceneActivated**

do in order

**this** myFirstMethod

**this** addTimeListener 1.0 add detail

declare procedure **timeElapsed** event getTimeSinceLastFire

do in order

drop statement here

**this** addKeyPressListener add detail

declare procedure **keyPressed** event isLetter event isDigit event getKey event isKey key:

do in order

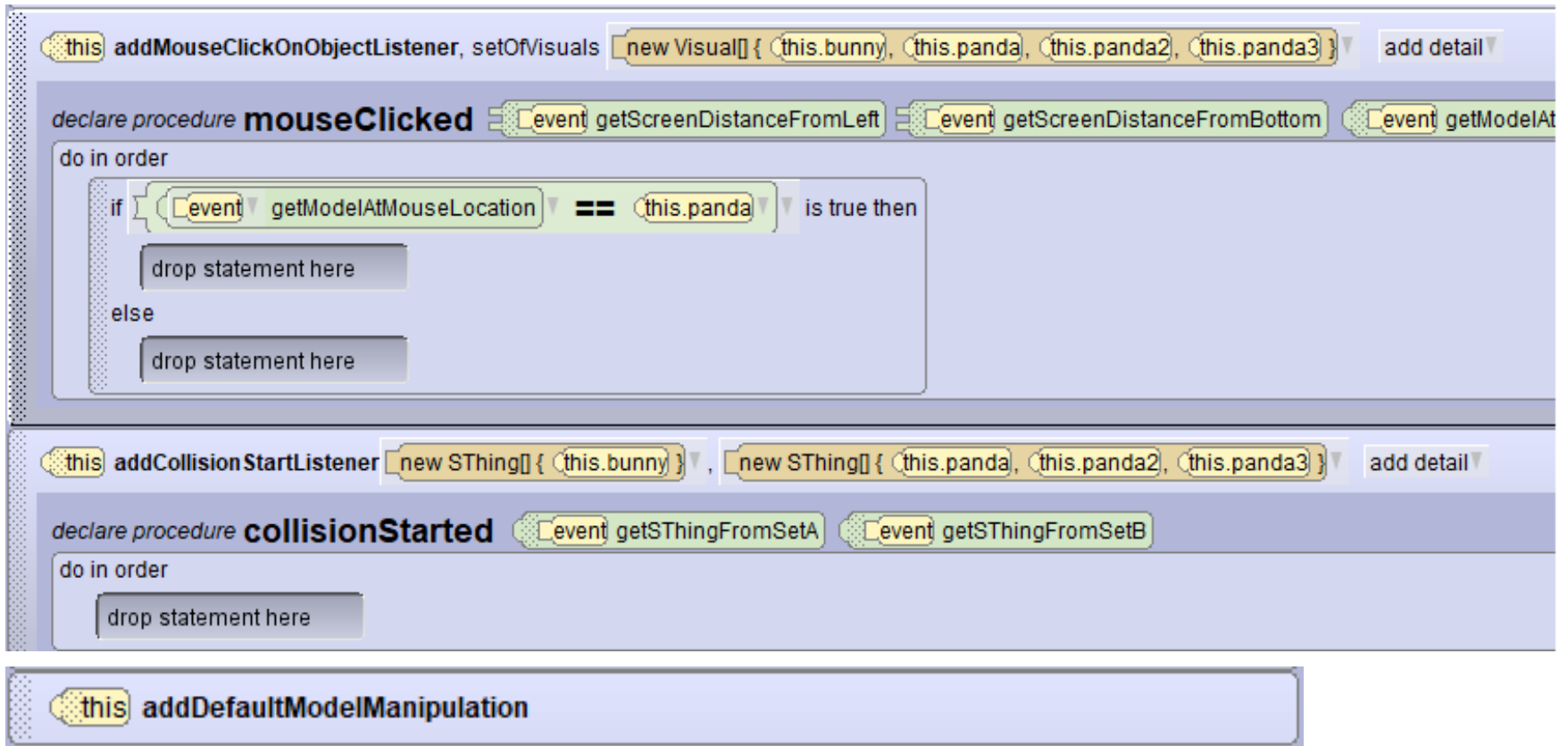
if event isKey S is true then

drop statement here

else

drop statement here

# Events 2



The image shows a sequence of Scratch code blocks. The first block is a 'this addMouseClickOnObjectListener' block with a 'setOfVisuals' field containing a list of 'this.bunny', 'this.panda', 'this.panda2', and 'this.panda3'. Below it is a 'declare procedure mouseClicked' block with three event inputs: 'getScreenDistanceFromLeft', 'getScreenDistanceFromBottom', and 'getModelAt'. The 'do in order' section contains an 'if' block where 'getModelAtMouseLocation' is compared to 'this.panda'. If true, it has a 'drop statement here' box; otherwise, it also has a 'drop statement here' box. The second block is a 'this addCollisionStartListener' block with two 'new SThing' fields: one containing 'this.bunny' and the other containing 'this.panda', 'this.panda2', and 'this.panda3'. Below it is a 'declare procedure collisionStarted' block with two event inputs: 'getSThingFromSetA' and 'getSThingFromSetB'. The 'do in order' section contains a 'drop statement here' box. The final block is a 'this addDefaultModelManipulation' block.

```
this addMouseClickOnObjectListener, setOfVisuals [new Visual[] { this.bunny, this.panda, this.panda2, this.panda3 } ] add detail ▼  
  
declare procedure mouseClicked [event] getScreenDistanceFromLeft [event] getScreenDistanceFromBottom [event] getModelAt  
do in order  
  if [event] getModelAtMouseLocation == [this.panda] is true then  
    drop statement here  
  else  
    drop statement here  
  
this addCollisionStartListener [new SThing[] { this.bunny } ], [new SThing[] { this.panda, this.panda2, this.panda3 } ] add detail ▼  
  
declare procedure collisionStarted [event] getSThingFromSetA [event] getSThingFromSetB  
do in order  
  drop statement here  
  
this addDefaultModelManipulation
```

# Events – when does it start, how does it work?

- sceneActivated
- addTimeListener
- keyPressed

# Events – when does it start, how does it work?

- sceneActivated
  - Starts when the world starts and executes all the code in it and then stops
- addTimeListener
  - Specify a time, such as 1.0 and then the event executes over and over, every 1.0 secs
- keyPressed
  - Every time you press any key or the particular key, the event starts executing



# Events – when does it start, how does it work? (part 2)

- addMouseClickedOnObjectListener
- addCollisionStartListener

# Events – when does it start, how does it work? (part 2)

- `addMouseClickedOnObjectListener`
  - Specify an array of objects that you can click on, then the variable *getModelAtMouseLocation* is the object you clicked on
- `addCollisionStartListener`
  - Specify two arrays, then whenever one item from one array collides with one item from the other array, then the event starts
  - Uses the variables: *getSthngFromSetA*, an object from the first array, and *getSthngFromSetB*, an object from the second array, such that these are the two objects that collided.

# Events – when does it start, how does it work? (part 3)

- defaultModelManipulation

# Events – when does it start, how does it work? (part 3)

- defaultModelManipulation
  - This lets you click on any object and drag it around.
  - Warning: You cannot guard this!

# How do you create a Scorer (or counter)

- A scorer/counter

# How do you create a Scorer (or counter)

- A scorer/counter
  - Need a 3D textModel (object)
  - Need a textModel property of type number
  - Update the number, then display it in the 3D textModel
  - Write procedures
    - initializeScore, updateScore

# How do you create A Countdown Timer

# How do you create A Countdown Timer

- Need 3D textModel (object)
- Need textModel property of type number
- Update the number by subtracting and then update the 3D text to display it
- Write Procedures:
  - InitializeTimer, UpdateTimer
- Need an addTimeListener Event
  - Will update every specified time unit
  - Need if, update only if game is on



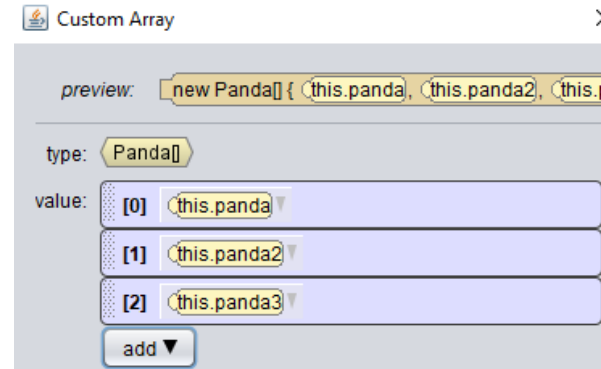
# Looping in Array – when and how to use each one

- For each in
- Each in together
- Indexing loop

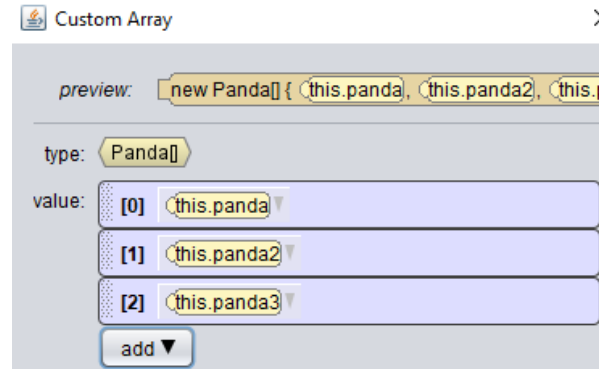
# Looping in Array – when and how to use each one

- For each in
  - Use with an array, to get each item in the array to do something one at a time
- Each in together
  - Use with an array, for each item at the same time to do something
- Indexing loop
  - Use when you need the *position* of array item
  - Use when need to change item in array
  - Use with count or while loop, use array.length
  - Create index variable, initialize it and update it

Q1) Given an array of pandas, how do I create an array of objectMarkers that match the pandas in the same order



Q1) Given an array of pandas, how do I create an array of objectMarkers that match the pandas in the same order



- Click on each panda in array in order and create object marker for it.
- Then create scene property, check array, and put object markers in the same order

# Q1) corresponding arrays

 Custom Array

preview: `new Panda[] { this.panda, this.pa`

type: `Panda[]`

value:

[0] `this.panda`

[1] `this.panda2`

[2] `this.panda3`

add ▼

 Custom Array

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

type: `SThingMarker[]`

value:

[0] `this.objectMarker1`

[1] `this.objectMarker2`

[2] `this.objectMarker3`

add ▼

unmanaged

▼ `Panda[]` **pandas** ← `new Panda[] { this.panda, this.panda2, this.panda3 }`

▼ `SThingMarker[]` **pandaObjectMarkers** ← `new SThingMarker[] { this.objectMarker1, this.objectMarker2, this.objectMarker3 }`

⊕ Add Scene Property...

Q2) Given pandas moved, what type of loop do you use to get them all back to their objectMarkers?



Q2) Given pandas moved, what type of loop do you use to get them all back to their objectMarkers?



An array index loop!

Q3) Given pandas moved, explain in words how to get them all back to their objectMarkers?





Q3) Given pandas moved, explain in words how to get them all back to their objectMarkers?

- Use indexArray Loop
- Loop over all the pandas
  - For the current panda, have it move And Orient To the panda Object Marker that is in the indexed position
  - Update the index

Q4: If we want to play a two level game, what might be the best game flow?

# Q4: Flow of game state for two level game

- Start in setup for game one
- Change to levelone – play game
- Change to setupLevelTwo
- Change to leveltwo – play game
- Change to gameover

# Class Today

- Build game with two levels

