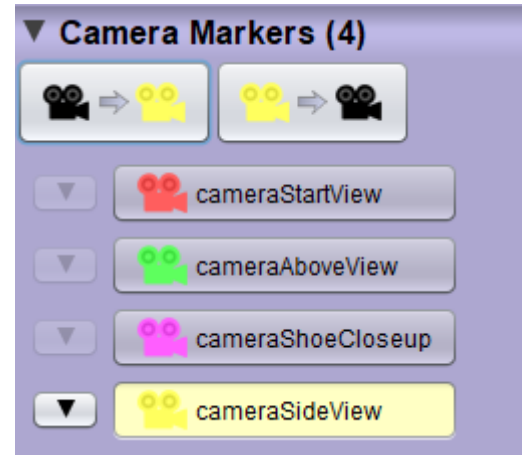


CompSci 94

Camera Controls

September 9, 2021



Prof. Susan Rodger

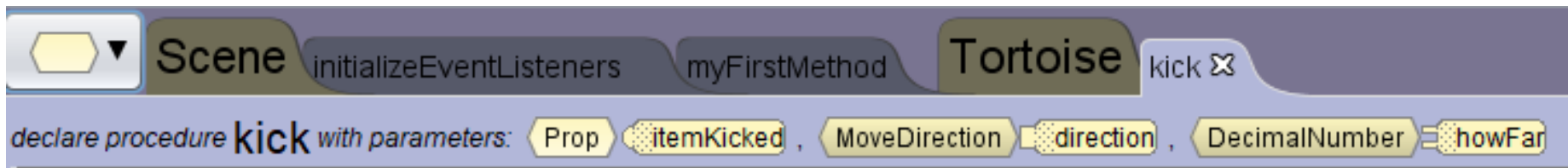
Announcements

- Assignment 2 due on Tuesday, Sept 14
 - Storyboard
 - Alice program
 - Make a Video – talk about code, run
- How to Submit Assignment 2
 - Storyboard, Alice program, video → Sakai
 - URL for Video → Reflect form
- Coming– procedures for classes of objects, and properties.
- QZ06 due Tuesday by class time

Review – Parameters/Arguments

Write the **tortoise kick** procedure

- It has three parameters
 - **itemKicked** of type **Prop** – the item to kick
 - **direction** of type **moveDirection** – the direction for the **itemKicked** object to move
 - **howFar** of type **DecimalNumber** – the distance for the item kicked to move



```
declare procedure kick with parameters: Prop itemKicked , MoveDirection direction , DecimalNumber howFar
```

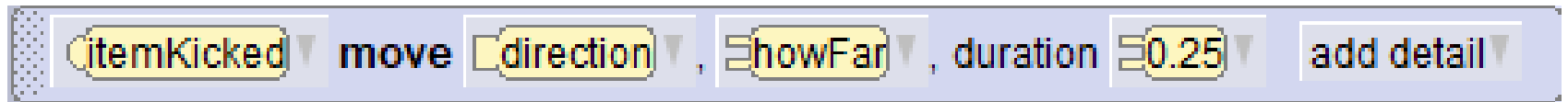
Review

Defining instruction, using parameters

- Defining the instruction



- Using the parameters in the kick code



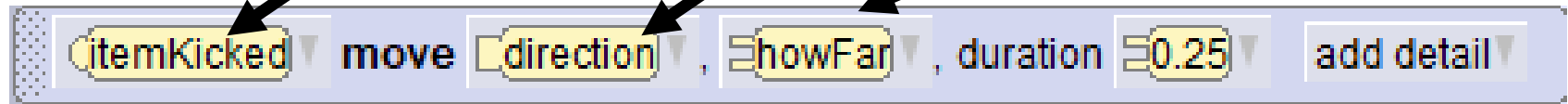
Review

Defining instruction, using parameters

- Defining the instruction



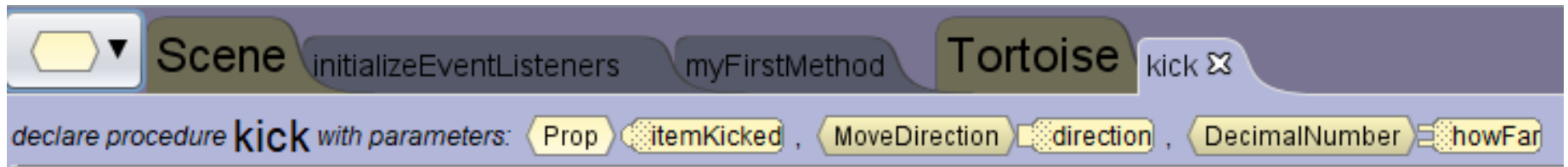
- Using the parameters in the kick code



Review

Defining instruction, Calling instruction

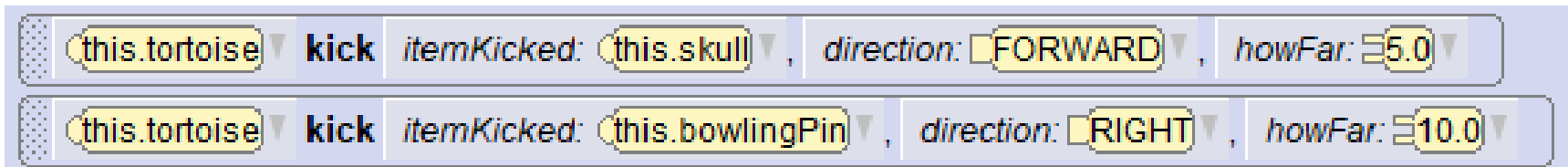
- Defining the instruction



The screenshot shows a code editor with two tabs: 'Scene' and 'Tortoise'. The 'Tortoise' tab is active and contains the following code:

```
declare procedure kick with parameters: Prop itemKicked, MoveDirection direction, DecimalNumber howFar
```

- Calling the instruction **in myFirstMethod**
pass arguments to parameters



The screenshot shows two lines of code in a code editor, each representing a call to the 'kick' procedure:

```
this.tortoise kick itemKicked: this.skull, direction: FORWARD, howFar: 5.0
```

```
this.tortoise kick itemKicked: this.bowlingPin, direction: RIGHT, howFar: 10.0
```

Review

Defining instruction, Calling instruction

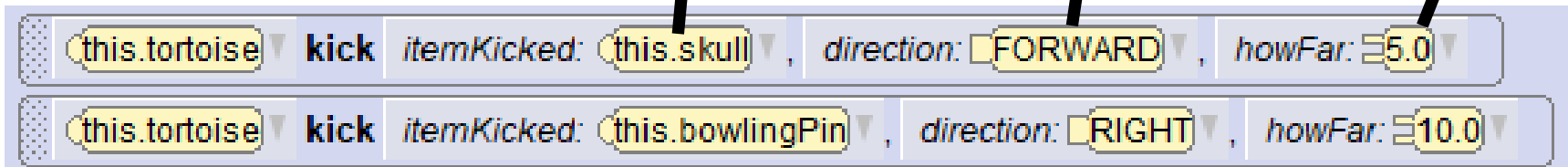
- Defining the instruction



```
declare procedure kick with parameters: Prop itemKicked, MoveDirection direction, DecimalNumber howFar
```

The screenshot shows the Scratch IDE interface. At the top, there are tabs for 'Scene', 'initializeEventListeners', 'myFirstMethod', and 'Tortoise'. The 'Tortoise' tab is active, and a 'kick' instruction is visible. Below the tabs, the procedure definition for 'kick' is shown: 'declare procedure kick with parameters: Prop itemKicked, MoveDirection direction, DecimalNumber howFar'. Three arrows point from the parameter names in the code below to the corresponding parameter names in this definition.

- Calling the instruction in myFirstMethod, pass arguments to parameters




```
this.tortoise kick itemKicked: this.skull, direction: FORWARD, howFar: 5.0  
this.tortoise kick itemKicked: this.bowlingPin, direction: RIGHT, howFar: 10.0
```

The screenshot shows two instances of the 'kick' instruction being called. The first instance is: 'this.tortoise kick itemKicked: this.skull, direction: FORWARD, howFar: 5.0'. The second instance is: 'this.tortoise kick itemKicked: this.bowlingPin, direction: RIGHT, howFar: 10.0'. Three arrows point from the argument values in these calls to the parameter names in the procedure definition above.

Review

Defining instruction, Calling instruction

- Defining the instruction



```
declare procedure kick with parameters: Prop itemKicked, MoveDirection direction, DecimalNumber howFar
```

The screenshot shows the Scratch IDE interface. The 'Tortoise' object is selected, and the 'kick' procedure is being defined. The parameters are: 'itemKicked' (Prop), 'direction' (MoveDirection), and 'howFar' (DecimalNumber).

- Calling the instruction in myFirstMethod, pass arguments to parameters



```
this.tortoise kick itemKicked: this.skull, direction: FORWARD, howFar: 5.0  
this.tortoise kick itemKicked: this.bowlingPin, direction: RIGHT, howFar: 10.0
```

The screenshot shows two instances of the 'kick' procedure being called. The first call uses 'this.skull' for 'itemKicked', 'FORWARD' for 'direction', and '5.0' for 'howFar'. The second call uses 'this.bowlingPin' for 'itemKicked', 'RIGHT' for 'direction', and '10.0' for 'howFar'. Arrows from the text above point to these arguments.

Call second time with different arguments!

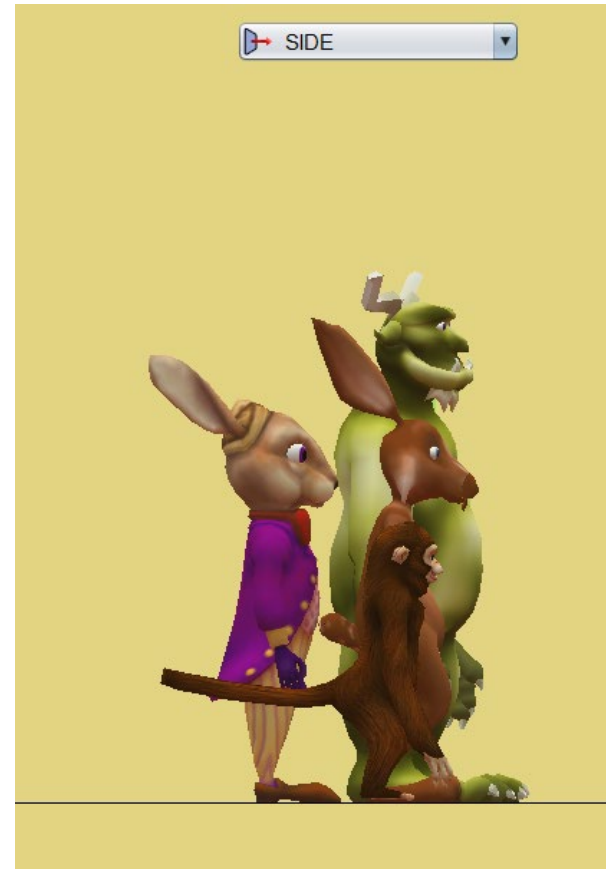
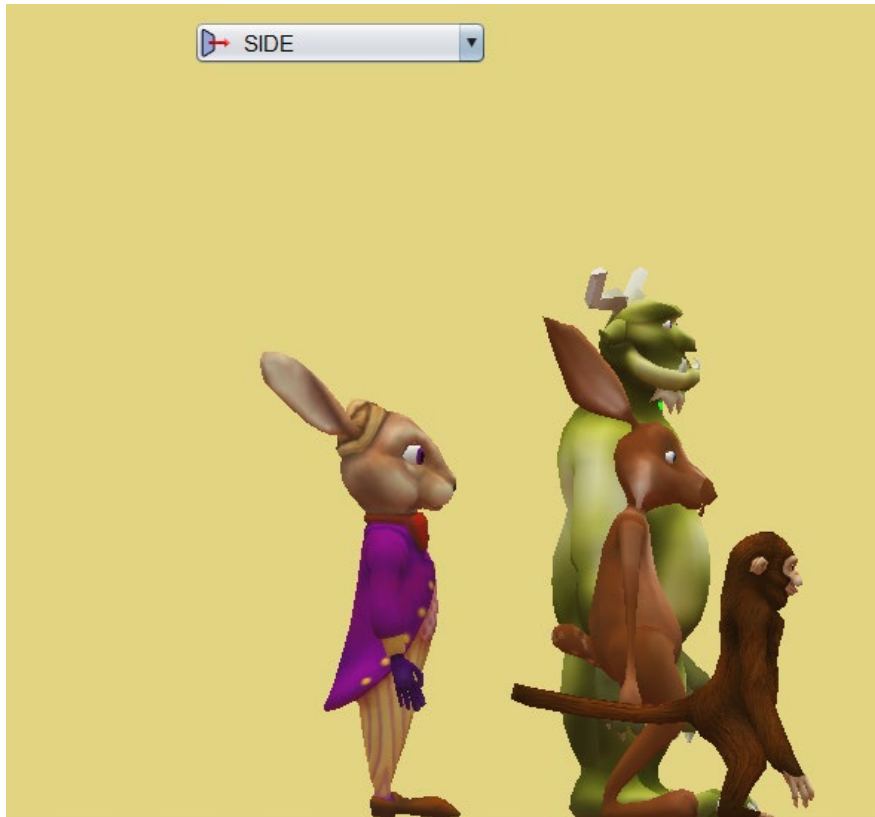
Q1. Camera Views

- How do you line up the animals in the front in a line?
- How do I make sure the animal behind the ogre is directly behind it?



Use 2D sideview for both

- Line up animals
- Move marchHare close to Ogre

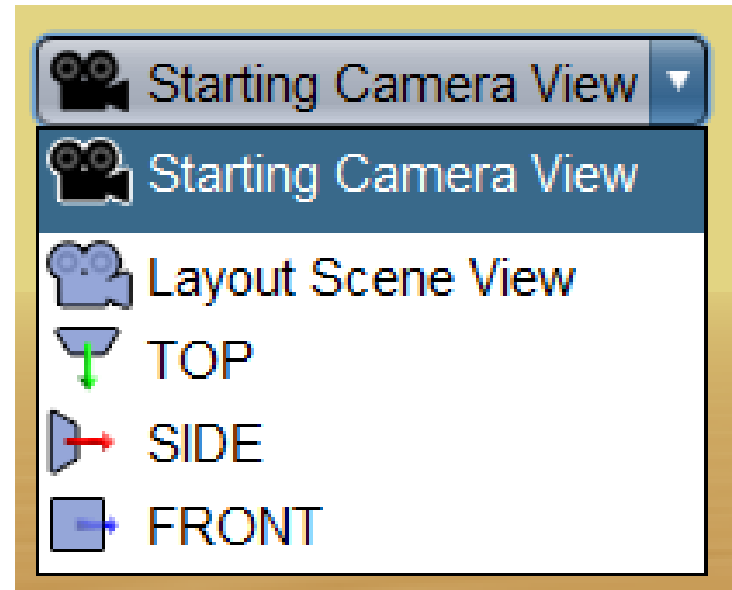


Q2. Setup Scene

- What other views can you use beside Starting Camera View?

Q2. Setup Scene

- What other views can you use beside Starting Camera View?
 - 2D TOP view
 - 2D SIDE view
 - 2D FRONT View
 - Layout Scene View



Q3. Camera Markers

- How does one create a camera marker?
- How does one use a Camera Marker during animation?

Q3. Camera Markers

- How does one create a camera marker?
 - Move camera to location.
 - Click on **add camera marker** in scene setup
 - Give camera marker a name.
- How does one use a Camera Marker during animation?
 - Use camera with **moveAndOrientTo** instruction to change a scene.

Q4. More on Camera markers

- When do you add Camera markers?
- What do these buttons mean?

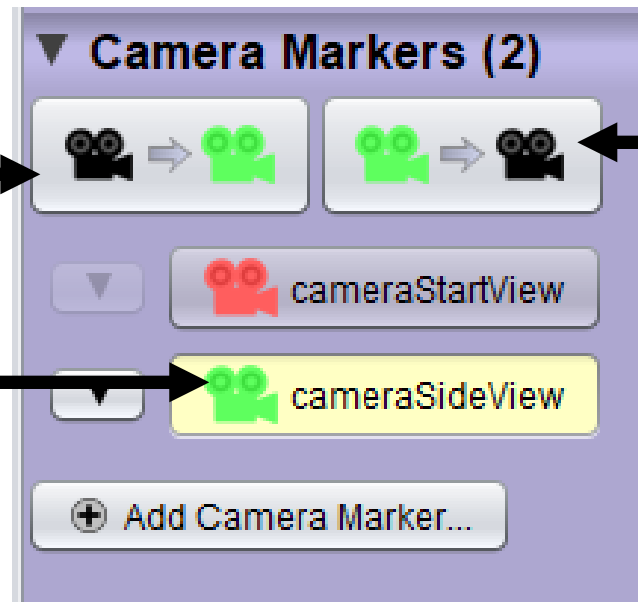
A)



B)



C)



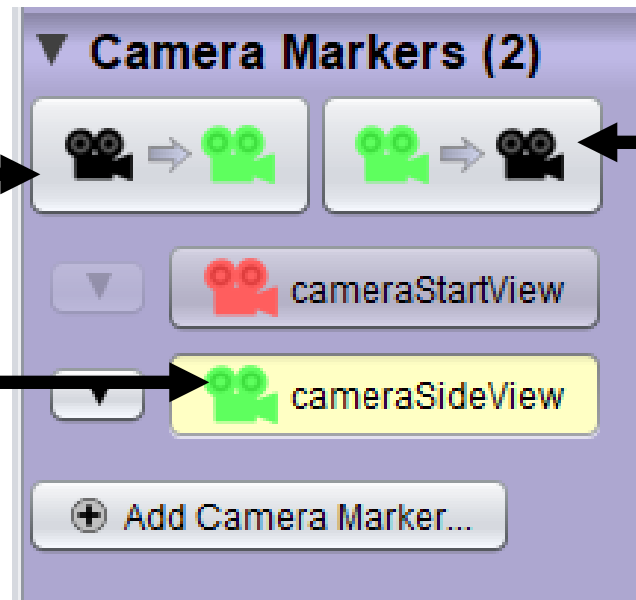
Q4. More on Camera markers

- When do you add Camera markers?
 - **LAST**, after the objects are placed
- What do these buttons mean?

A) Move camera to
Camera marker

B) Move camera
marker to
camera

C) Select camera
marker



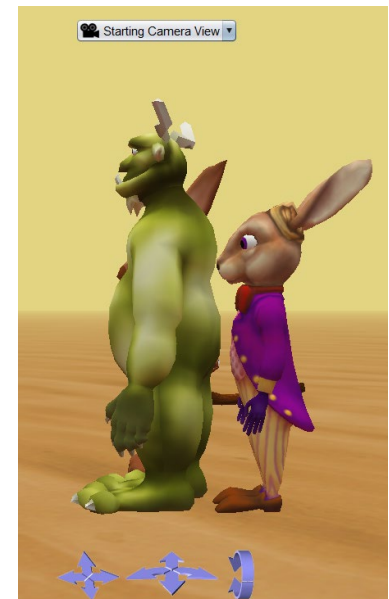
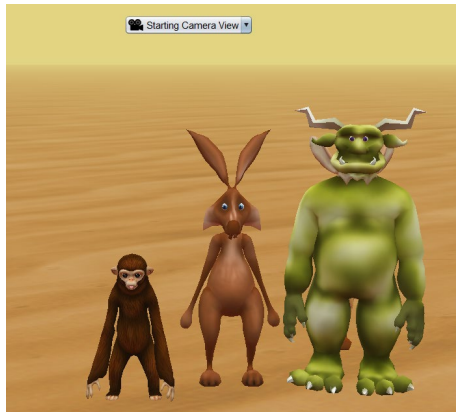
Q5. Setting up a CameraMarker

- How does one setup camera for side view?
 - (give all the steps you would do)



Q5. Setting up a CameraMarker

- How does one setup camera for side view?
 - Using one-shots, have camera **move to** hare.
 - Then camera **orientToUpright**
 - Camera move up 1.0
 - Camera move right 6.0
 - Camera turn left 0.25
 - Use purple arrows to adjust view.



Class Today

- Continue writing procedures with parameters
- Moving between camera views

