CompSci 4
Chap 6 Sec 1
September 22, 2009

Prof. Susan Rodger
Announcements

• Exam 1 Thursday, Oct 1
  – Closed book, closed notes, closed neighbor
  – Chaps 1-2, Chaps 4, 6, html
  – On Thursday, will give you an old exam to work on, then review it next Tuesday

• Assignment 4 storyboard due Thursday
  – Alice world due Thursday too
What we will do today

• Lecture on Chap 6, Sec 1 - Functions
• Classwork
Functionality

• A function
  – Receives value(s)
  – Performs computation on value(s)
  – Returns (sends back) a value
Types of functions

• The type of a function depends on the type of value it returns
  – a calculated value (a number)
  – a specific object
  – a color
  – etc.
Built-in functions

• Used one of Alice’s built-in functions
  – `skateAround` method for the `cleverSkater`

• Let’s look at another example.
Example

• Move ball to within 1 meter of net, then bounce ball over the net.
  – Bounce - Ball should move up and forward, then down and forward
Move Ball to 1 meter from Net

- Use “distance to” function and math
Height

- Can use the built-in height function to determine the height of the net and move the ball up that distance

Demo – what happens?
Rolling the ball

- How do we roll the ball along the ground?
- Want a realistic motion rather than a slide
- The ball must simultaneously move and roll.
- The ball must roll “as seen by” ground
- The ball and ground must face the same direction
Demo: A first attempt

```
toyBall1.test No parameters

No variables

- toyBall1 turn to face tennisNet more...
- ground turn to face tennisNet more...

- Do together
  - toyBall1 move forward 2 meters asSeenBy = ground more...
  - toyBall1 turn forward 2 revolutions more...
```
Revising the Approach

• The ball is made to roll 1 revolution.
• What if we want the ball to roll a certain distance?
• How can we make the ball roll the correct number of revolutions to cover a given distance along the ground?
Number of Revolutions

• The number of revolutions depends on the size of the ball
  – The number of revolutions is\( \text{distance} / (\pi * \text{diameter}) \)

• There is no built-in function to return the number of revolutions
  – Must write it!
Parameters

• We want to return the value computed as
  \[ \text{Distance} / \pi \times \text{diameter} \]
  where \( \pi = 3.14 \ldots \)

• Obviously, what is needed
  – The ball’s diameter
    • The ball object has a built-in width function
  – The distance the ball is to travel
    • Can be sent as a parameter to the function
numberOfRevolutions function
Demo: Calling the function

This is a test value

- Run the animation with several test values
- Make sure it works as expected
- What happens if you use a negative value?
- Add a parameter for distance
Now Ball roll to net?

- Difficult….
- ToyBall turn to face TennisNet and roll, what happens?
Tricky – Orient To

- `toyBall1`: orient to `ground` more...
- `toyBall1`: turn to face `tennisNet` more...
- `ground`: turn to face `tennisNet` more...
- `toyBall1` realisticRoll `distance`:
  - `{ toyBall1: distance to tennisNet - 1 }`
- `toyBall1`: orient to `world` more...
- `toyBall1`: turn to face `tennisNet` more...

**Do together**
- `toyBall1`: move forward 2 meters more...

**Do in order**
- `toyBall1`: move up (subject = tennisNet's height * 1.5)
- `toyBall1`: move down (subject = tennisNet's height * 1.5)
Levels of functions

- As with methods, you can write functions as either class-level or world-level. (what was the function we just wrote?)
- Guidelines for class-level methods apply to class-level functions:
  - No references to other objects
  - No references to world-level functions
  - Built-in world-level functions are ok
Classwork today