UNIVERSITY OF
CALGARY

CPSC203 – Introduction to Problem Solving and Using Application Software

Winter 2010

Tutorial 8: Mehrdad Nurolahzade

Introduction

- Dummy Objects
- Method Parameters
- Adding Parameters to Existing Methods
- Reusing Modified Classes

Progress Reports

- The 2nd and 3rd progress reports will be submitted through the Blackboard.
- Progress reports are expected from individual team members NOT teams.
- No marks is going to be deducted if you did not submit the 1st progress report.
- Missing the 2nd and 3rd progress reports will result in deduction of 5 marks EACH.

Alice Resources

- I have added some Alice video tutorials and sample student projects to the TA Examples for Alice Programming page of the course Wiki: http://wiki.ucalgary.ca/page/Courses/Computer_Science/CPSC_203/CPSC_203_Template/Winter_2010_Lab_Manual/TA_Examples_for_Alice_Programming
- Lot's of good ideas, inspirations, tips and tricks for your final project in there. DON'T MISS IT!

Dummy Object

- To shift the camera's perspective back to the original view.
- Since there are no objects where the camera originally was, we cannot move the camera back to that perspective.
- For this reason, Alice allows you to place dummy objects where we want to move the camera.

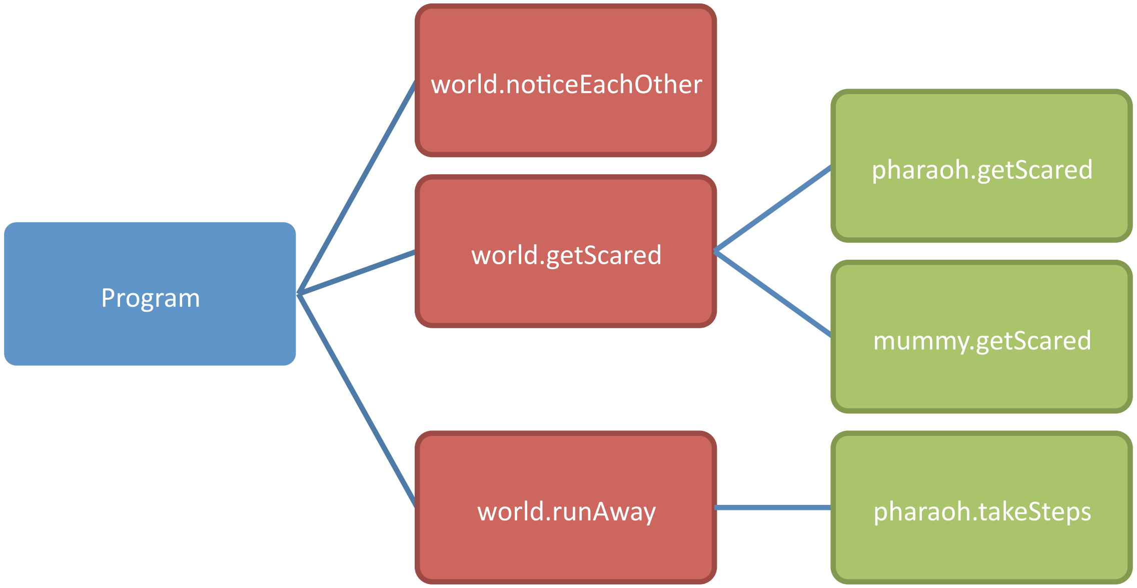
Method Parameters

- Parameters allow you to send information to methods.
- Most of the built-in methods you used so far required parameters.
- For instance, the turn method requires two parameters: the direction and the amount of turning.
- We can also add parameters to the methods we create.

pharaoh.takeSteps

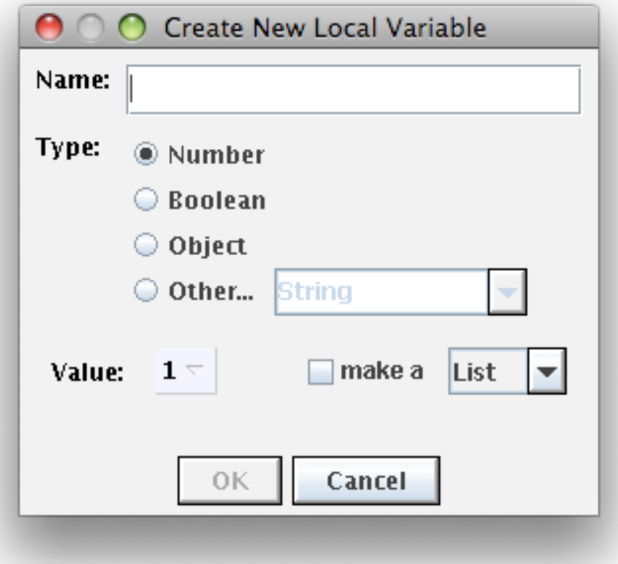
- Move the large do together loop (using the clipboard) into a new class-level method for the *pharaoh*. Name the method *takeSteps*.

- Call *takeSteps* from inside the loop in the *world.runAway* method.



Number of Steps Parameter

- Let us further improve the *takeSteps* method so that the number of steps is a parameter.
- From the *pharaoh.takeSteps* method, select the “create new parameter” button.
- Parameters can have numerous types. Obviously, the type we need is Number.
- Set the name of the parameter as *numberOfSteps*.



Create New Local Variable

Name:

Type: ● Number ○ Boolean ○ Object ○ Other... String

Value: 1 □ make a List

Number of Steps Parameter

- Now we want to repeat the code in the method a *numberOfSteps* times.

- To repeat the loop *numberOfSteps* times, drag the parameter *numberOfSteps* to set the loop repetition times.

Exercise

- Add another parameter to the *pharaoh.takeSteps* method.
- Name the parameter *timePerStep*.
- Each of the steps taken should take that amount of time of *timePerStep*.
- Don't forget to set the duration of the individual limb movement using expressions and maths.

Exercise

- Modify *pharaoh.takeSteps* so that it takes a third parameter of type object.
- The *pharaoh* must run towards that object.
- Test your method using the *pyramid* and *sphinx* as different test scenarios.

Reusing Modified Classes

- Now that we have added new methods to the *pharaoh* class, it seems like a shame to not be able to use our new methods in different worlds (i.e. future programs).
- What we could do is save our version of the pharaoh as a new class, with a new name.

- This new class inherits all the original methods, properties and functions of the original *pharaoh* class, but also contains the new methods we defined.

Renaming and Saving Classes

- Rename the class: Right-click the object you wish to rename and select rename from the menu. Let us rename the class to *ourOwnPharaoh*.

- Save the new class: Right-click on *ourOwnPharaoh* and select Save Object. Select the Save button in the dialog box that appears.

Importing Classes

- The next time we need to use *ourOwnPharoah* in a world, we simply select Import from the File menu, and browse for the class we created.