Comp 401 – Assignment 8: Complex Object Interaction

Date Assigned: Wed Oct 19, 2011
Completion Date: Fri Oct 28, 2011
Early Submission Date: Wed Oct 26, 2011

So far, there has been little interaction between different objects. Most of it has been between an object and its sub-objects - when an object moves, it asks its descendants to also move. In this assignment, you will program more complex object interaction. You will also define another collection type for the scanner part of your assignment. This part is optional, and you can postpone it to next assignment. You should have enough time to complete the assignment by the completion date even if you do not work during the October break.

**Token List (Optional)**
Define a class (and interface) to represent a collection of token-objects (which were defined in the token assignment). For now, make the collection a history. Later you might need to modify the collection to implement some extra credit features.

**Input String with Token History (Optional)**
Create another version of the stateful scanner class of assignment 7 (which built on the versions in assignment 4 and 3). Like this class, the new class has two properties. One of these properties is the independent editable property storing the scanned string. The second property, however, is no longer a string property. Instead, it is a dependent readonly property storing an instance of the token history class that contains all the tokens in the scanned string (in the order these tokens were scanned). You cannot subclass the stateful class of assignment 7, as a subclass cannot hide a property defined by the superclass. Simply create a new version of this class by copying and then editing the original code.

For extra credit, reuse the code in the previous version to write the new version (without using inheritance).

**Implement Optional Part of Last Assignment**
Assignment 7 had the following optional step:

Replace components of the scene of the previous assignment (assignment 6) with the new versions of the two avatar collections and the guard (assignment 7). Also add the fallen knight collection to the scene.

Implement this step.
Global Class
Create a class defining an editable static property storing the Y position of the next chat message. The initial value can be a value chosen by you. Define also a method to decrement the property by some constant value chosen by you. This property will be used to determine where the next chat message will go.

Simulation Methods
Your simulation will now come alive with methods that involve interaction among its objects. It should define methods to:

1. Approaching knights moves: Move the approaching knights by a certain int distance in the x and y directions.
2. Leading knight crosses bridge: Move the second knight (if such a knight exists) to the position of the first knight (if such a knight exists), remove the first element of the approaching knights collection, and add the removed knight to the last element of the departing knights collection, (while, of course ensuring the alignment constraints of the two collections). This method will take no parameters.
3. Leading knight falls down: The same as above, except that the removed knight goes to the fallen knights collection and its position is set to coordinates chosen by you.
4. Leading knight speaks: Add a chat message to the chat history of the first knight in the approaching knight collection (while of course maintaining the alignment constraint). This method takes a string as an argument and converts it into a corresponding graphic object accepted by the chat history. The Y position of the label is determined by the static property. This method should also decrement the static property for the next chat message in the knight or the guard.
5. Guard speaks: Same as above, except that the message is added to the chat history of the guard.
6. Guard moves: Move the guard (with the chat history) a certain int distance in the x and y position.

Advanced Simulation Methods (Optional Part)
Use these operations to define additional operations that (a) ensure that the leading knight cannot cross the bridge unless three questions have been answered or the guard has moved away from its standing area, and (b) a knight can speak only if it is in the knight standing area (b) the leading knight and guard alternate speaking.