Exercise – Agents

Exercise 1:

Based on the lecture notes, implement a **GameObject** class to represents Agents. The Agent should have a virtual update / draw function.

Although GameObjects could be anything, for now we will include the following variables and properties:

* Position (i.e. float x, y, or Vector2 position)

Exercise 2:

Create an IBehaviour class, such as that displayed in the lecture. Include a virtual execute() or update() method which takes a GameObject pointer and deltaTime as parameters.

Update the GameObject class to contain a collection of IBehaviour pointers. In the GameObject’s update, iterate over the IBehaviours and call the IBehaviour’s execute()/update() method.

Although for now we have no derived behaviours, we will shortly.

Exercise 3:

Finally, let’s create some behaviours. Implement the following classes derived from IBehaviour:

* KeyboardControler behaviour.  
  This should override the update function and move the GameObject passed in as a parameter, based on keyboard key presses. For example, the KeyboardController might have a speed property, and during the update it might use the keyboard arrow keys to move a 2-dimensional GameObject by adding a direction, multiplied by deltaTime and speed, to the GameObject’s position.
* FollowGameObject behaviour.  
  This should override the update function and move the GameObject agent parameter towards a target GameObject. The behaviour should have a GameObject as a target, and should include a speed property much like the Keyboard Controller behaviour above. Using the speed and deltaTime you would move the agent towards the target each frame.

Exercise 4:

Create an application that contains 2 GameObject instances, one which has a KeyboardController behaviour added to it, and another that has a FollowGameObject behaviour added to it where the target GameObject is the one using the KeyboardController.