Comp 110-003 - Assignment 9:
MVC and Animation

Date Assigned: Mon Nov 28, 2011
Completion Date: Mon Dec 7, 2011 (midnight)

In this assignment, you will strengthen your knowledge of the MVC programming paradigm and get practice with animations. As in the previous two assignments, you will continue to improve your structured object programming skills.

Start this assignment with all of your code from the previous assignment!

Part 1: Turning Rectangle, Oval, and Highway into observables
Extend your Rectangle and Oval classes from the previous assignments by making them an observable that follows the Java Beans observable pattern. There are three steps that you must do:

1) Define a PropertyChangeListener history
2) Implement the standard addPropertyChangeListener method in AHighway
3) After you change a value of one of the Rectangle/Oval properties, notify all observers

For help on completing these steps, read the “Variations in Observer/Observable Communication” section in the MVC chapter notes. You can find additional examples in the AnAnimatingShuttleLocation discussion in the Array chapter notes.

Recall that the object that is actually changed is the one that must notify the observers. For example, when a car location is modified, it should be Rectangle and Oval instances making up the car that notify observers.

You do not have to implement PropertyChangeEvent and PropertyChangeListener. Instead, you can need to import them from the java.beans package. To do so, include the following two lines in every class that uses PropertyChangeEvent and PropertyChangeListener instances:

```java
import java.beans.PropertyChangeListener;
import java.beans.PropertyChangeEvent;
```
Part 2: Moving the rabbit in all directions

Extend your AHighway by adding a moveRabbitLeft, moveRabbitRight, moveRabbitUp, and moveRabbitDown operations, which move the rabbit left, right, up, and down, respectively, by RabbitMoveDistance (defined in previous assignment). This should not take long because you have already defined moveRabbit in the previous assignment, which moved the rabbit vertically by MoveRabbitDistance. As was the case with moveRabbit in the previous assignment, each one of the new move operations must also check if the rabbit is colliding with any of the cars and update the RabbitStatus property accordingly.

If in the previous assignment, you moved the cars or the rabbit whenever the corresponding move distance properties changed, you will have to correct your assignment. The rabbit and the cars need to move when you or a user (which can be another object) calls moveAllCar or moveRabbit(Left/Right/Up/Down).

Also, if in the previous assignment, you used the move distance property values as offsets, you will need to correct your assignment. These properties stored by how much the cars and the rabbit should move, not to what location they should move.

Part 3: AConsoleHighwayController

Implement a console-based controller for AHighway, called AConsoleHighwayController. It should implement the following interface:

```java
public interface HighwayController {
    public void setModel(Highway model);
    public void processInput();
}
```

The processInput command accepts user input from the console. Assume the user can enter the following commands only (hence, no erroneous input checking is required):

1) ‘a’ to move the rabbit left
2) ‘d’ to move the rabbit right
3) ‘w’ to move the rabbit up
4) ‘s’ to move the rabbit down
5) ‘q’ to stop entering commands

Each command must be followed by Enter. This is a somewhat awkward input interface, but it makes the code you have to write a little less complex. So to move the rabbit up twice, the user would type first w, then Enter, then w again, and then Enter again.
Part 4: Animating AHighway
Define two editable integer properties, AnimationDistance and AnimationPause, in AHighway. Then implement a startGame operation in AHighway that has an infinite loop that does the following:

1) Move the cars by AnimationStepSize to the left
2) Sleep for AnimationPauseTime time

NOTE: After a while, all of the cars on the highway will move off the screen. This behavior is acceptable.

Part 5: AHighwayDriver
Implement your main method in AHighwayDriver. The method is responsible for:

1) instantiating AHighway (the model)
2) instantiating AConsoleBasedHighwayController (the controller)
3) Connecting the model and the controller
4) Calling bus.ugen.ObjectEditor.edit to display the highway

AHighwayDriver also sets up the highway for the game as follows:

1) Sets MoveCarDistance to 10
2) Sets PreviousCarDistance to 150
3) Sets MoveRabbitDistance to 35
4) Sets AnimationStepSize to 10
5) Sets AnimationPauseTime to 30
6) Adds 30 cars to the highway, each of which is of width 100 and height 30
Playing the Game

1) The user first runs AHighwayDriver. The following screen should appear.

2) Once the highway is displayed in an ObjectEditor window, the user invokes the StartGame operation on the Highway as shown below.

3) Then the user enters commands to move the rabbit.
**Finishing the Game**

There are two ways to finish the game.

1) Rabbit crosses the road without getting hit: In this case, you need to display a “Congratulations! You live to play another day” message in a JOptionPane.

2) Rabbit gets hit: In this case, you need to display a “Splat! Please try again” message in a JOptionPane.

When a finishing condition is reached, you do not have to stop the cars from moving. The reason for the latter is that in the real-world, cars on a highway will not stop because of a rabbit. Also, assume that the user will not enter any more commands once the finish message is displayed. In other words, you do not have to write any code to explicitly prevent the user from entering anything but a non ‘q’ command.

**Bonus**

For the past few months, the bunny world has been hopping. A rumor has been spreading that the legend of Bravehops will be fulfilled on Dec 5, 2007. For those of you who have not heard of Bravehops, it is a legend of a rabbit who escapes the Land Below the Highway and reaches the world beyond. Rabbits from the farthest reaches of the Land Below the Highway have gathered in the grassy area just inside the border of their world in anticipation of the fulfillment of the legend. If Bravehops manages to reach the other side, they will all stand up and clap their ears. But if Bravehops does not reach the other side, they will quietly leave without ever being seen.

Your bonus, should you choose to complete it, is to draw the spectators once Bravehops crosses the highway. The number of spectators you need to draw the n\(^\text{th}\) number in the Fibonacci sequence. The value of n is stored in an editable integer property of AHighway, called FibonacciNumber. You must use recursion to calculate the number of spectators from the FibonacciNumber value. The n\(^\text{th}\) number in the Fibonacci sequence is calculated as follows:

\[
F(1) = 1 \quad \text{if } n = 1 \\
F(2) = 1 \quad \text{if } n = 2 \\
F(n) = F(n-1) + F(n-2) \quad \text{if } n > 2
\]

While the exact locations of and the separation between the spectators is not important, an outside observer should be able to count the exact number of spectators. In other words, don’t draw them all in the same location because it will appear to an outside observer that there is only one spectator regardless of how many you actually draw.