CSE1720

Week 10, **Class Meeting 27** (Lab 10)

Winter 2013 ◆
Thursday, Mar 15, 2013 & Friday, Mar 16, 2013

YORK UNIVERSITÉ UNIVERSITY

Week 10 Lab Exercise

- **Due Date:** Monday, Mar 18, 2013, 11:59pm
- **Course Weight:** 2.0%
- **Topic**: Adding features to your game and customization

```
submit 1720 W10Lab10 screenshot
```

See Exercise #5 for instructions on what to submit.

Setup

Download the latest version of the **code base** from the course website (lab sub-page)

The classes are modified in several ways.

Exercise #1: Fixing Target Hitting

Run the app by invoking `GameDriver` in the default package.

Did you notice from the last lab exercises that there was a problem with the positioning of the target sprites?

Specifically, when the targets were placed toward the edges of the screen, the shooter could not be positioned close enough to the edge to launch projectiles that could actually shoot the target.

We will fix this next…

Exercise #1A: Fixing Target Hitting

Approach #1

- Examine the method `getRandomPositionAtTopOfScreen()` in the `TargetSprite` class.
- Locate the expression that assigns the value to `x`. Create a copy of the statement and comment it out. Experiment with different values (e.g., set the value to `0` and/or `width`.) Notice with these values, the targets cannot be hit.
- Ideally, the method would query the size of the ShooterSprite and use that value as the basis for the distance between the left and right boundaries and the target position. But the ShooterSprite is not visible to this class!
- The method uses the *diameter of the target* as the basis for the distance. But this is not a good solution.
- **TASK:** Come up with another solution!

Exercise #1B: Fixing Target Hitting

Approach #2

- Look at the `moveRight()` and `moveLeft()` methods of `BasicTriangleShooter`. Understand why the movement of the shooter is blocked with it reaches the right or left edge.
- **TASK:** Change the behaviour of the `BasicTriangleShooter` shooter sprite so that when it reaches the right or left edge of the screen, the sprite “wraps around” to the other side!

Exercise #2: Keeping Track of Ammo

Have a look in in the class `ScoreTallySprite`. See that there is now a method `registerAmmoUsed()`.

Have a look in the `SpriteDataModel`. See that every time the user fires the shooter (the method `fireShooter()` is invoked), the `ScoreTallySprite` is mutated. This signals that another shot has been fired.

TASK: Modify the way that the `ScoreTallySprite` is drawn so that the user can see how much ammo he or she is using.

FURTHER TASK: modify the scoring to reward the user for lower use of ammo and/or penalize the user for using too much ammo.

Exercise #3: Different Ways to Move

Have a look in in the class `BasicTriangleShooter`. See that there are now methods called `rotateLeft()`and `rotateRight()`.

TASK: Modify the `Controller` class so that the left and right cursor keys serve to rotate the shooter (rather than move it left/right)

FURTHER TASK: customize the control of ShooterSprite is an interesting new way.

FURTHER TASK: notice how the projectiles sometimes don't reach all the way to the target. Modify the lifetime of the projectiles to change their behaviour.

Exercise #4: Using a Different Shooter

Now we want to modify the `SpriteDataModel` class.

Notice the statement in the constructor:

```
theShooter = new BasicTriangleShooter(dimensionAvailable);
```

TASK: In the place of `BasicTriangleShooter`, use the services of other `ShooterSprite` class (which is called `FroggyShooter`).

Notice how this design works. The code base uses the `ShooterSprite` interface to declare the variable `theShooter` and an implementing class, `BasicTriangleShooter`, for the RHS to construct the object.

Exercise #5: Start Customizing

You should now start customizing the game however you like. You can copy the `FroggySprite` class and swap in whichever images you like.

If you like, you can introduce other images for the Targets, instead of the round circles.

You can tailor the way the game is scored (add timing and/or ammo penalties).

Make these modifications and run the game. Capture a screen shot of the game that shows off the customizations that you have done. For the screen shot, use one of the common graphics formats, such as (jpg, tif, gif).

```
submit 1720 W10Lab10 screenshot
```