CSE1720

Week 05, **Class Meeting 15** (Lab 05)

Winter 2013 ◆
Thursday, Feb 7, 2013 & Friday, Feb 8, 2013

Week 05 Lab Exercise

- **Due Date:** Monday, Feb 11, 2013, 11:59pm
- **Course Weight:** 2%
- **Topic**: Implementing a collection

Once you have completed the exercises, submit the following:

```
submit 1720 W05Lab05 GameCanvas.java
submit 1720 W05Lab05 Sprite.java
submit 1720 W05Lab05 ShooterSprite.java
submit 1720 W05Lab05 ProjectileSprite.java
```

Exercise #1

Examine the game code base

Create a new interface called `Sprite`

(File ➔ New ➔ Interface)

In that class' empty class body, place the following line:

```
	public void specifyDrawing(Graphics2D g);
```

Now change the class header of the two sprite classes to implement this interface, like this (and analogously for the other sprite class)

```
public class ShooterSprite implements Sprite {
```

Run the app and get convinced that these modifications have not changed the app's functionality in any way

3

Exercise #2

Now, have a look at the class `GameCanvas`, in which the two sprite objects are employed.

Declare a third class attribute to represent a collection of elements of type `Sprite`. In the constructor, and instantiate the collection.

Here is the declaration

```
Collection<Sprite> theSprites;
```

Here is the instantiation:

```
theSprites = new LinkedList<Sprite>();
```

Locate these statements in appropriate places.

You will need to add appropriate import statements.

4

Exercise #3

Populate the collection with the sprite objects.

You need to locate the correct places in the class for doing this.

5

Exercise #4

Locate the method that is named below.

```
public void drawOnScreenElements(Graphics2D g)
```

Note that in the body of this method, each sprite is redrawn individually.

Once we add more and more sprites, this approach will get very cumbersome. Instead, let's use a better approach.

Modify the method so that it iterates over the **collection** of sprites. For each sprite, invokes the `drawOnScreenElements` method on it.

It is fine to test whether each sprite is non-null before method invocation.

6

Postscript

You should stand in awe at the total coolness of this approach.

The collection contains different types of sprites, and each sprite has its own way of specifying how it should be drawn.

With this approach, you can iterate over the collection and invoke `drawOnScreenElements(Graphics2D g)` for **any** element in the collection and **the appropriate version** of the method will get invoked.

This is the beauty of polymorphism, which is described in Chapter 9.

Postscript, part 2

The use of a collection also takes care of a limitation of the game, namely that only one projectile could be on the screen at once.

Can you diagnose why this was, and how the collection approach solved the problem?