Computer Science 62

Assignment 1

due at 11:59 pm on Tuesday, February 2, 2009

Our first assignment is a review of Java, an exercise in building classes, and an illustration of the `ArrayList` class. You will create a graphical version of Bailey's Silver Dollar Game.

You should have done this in the lab already, but make sure you understand the instructions for submitting assignments, linked under the course Resources section, and make sure you've setup `Eclipse` appropriately (see Lab 1).

1 The Problem

1. Read through the Java style guide linked on the course web page under Resources. You will need to follow these guidlines for all of your assignments.

2. Begin by creating a new Java project in Eclipse named `Assignment1`. (Remember to add the `BAILEY` variable.)

If you ever need to add a library variable (i.e. `BAILEY`) to an existing project, for exmple, if you click finish too quickly :), right-click on a project (on a Mac, that's `ctrl`+click) and select `Build Path/Configure Build Path...`. Select the `Libraries` tab and then click `Add Variable...`

As in Lab 1, to get started, copy the files over into the `src` directory of your newly created project using the Terminal. There are three files `Coin.java`, `CoinSquare.java` and `GraphicsCoinStrip.java` all of which can be found at `/common/cs/cs062/assignments/assignment1/`. Use the `cp` command like before. As a shortcut, you can specify all of the files in a directory when copying using `*`, for example `/common/cs/cs062/assignments/assignment1/*` would refer to all of the files for assignment 1.

Once you've copied them, refresh your project in eclipse. The classes `Coin.java` and `CoinSquare.java` are complete; you will not have to change them, but take a look at them to see what methods are available.

3. In the file `GraphicsCoinStrip.java`, you must fill in the constructor and add appropriate methods where necessary to play the game. Follow the pattern from `TextCoinStrip` in our first laboratory session. Notice that there are no `play` or `move` methods, because the mouse is in control of the game.

Much of what drives the game is the mouse event handling, which can be found in `CoinMouseListener` in the middle of `GraphicsCoinStrip.java`. I have given you some of the code that should be there as well as some comments suggesting what you need to add. The `contains` methods that `Coin` and `CoinSquare` inherit from `Elipse2D` and `Rectangle2D` may be helpful.

As much as possible, try and develop incrementally. Get one small piece working and then move on to another chunk.

4. After you have a working copy of the game, write a method that checks to see if the game has completed and signal this to the user in some fashion. Possible examples are to print out a message to the console, or better, change the color of all of the coins. It is not required, but you can also make sure that the coins no longer move once the game has completed.

5. When you are finished, submit *only* your file `GraphicsCoinStrip.java`.

2 Commenting

As always, you should comment your code. In addition, we will be using Javadoc commenting style. You *must* have the following to be compliant with Javadoc:

- A description of the class at the beginning
- `@author` after the class description
- `@date` after the author tag

- A description of each method before the method
- `@param`, `@return` and `@throws` tags for each method (where appropriate)
- Appropriate Javadoc "style" as we discussed in class (that is they should start with /**, have a * before each line and be indented appropriately)

You *may* also want to include `@pre` and `@post` tags for the methods, but they are not required.

3 Grading

You will be graded based on the following criteria:

criterion	points
game starts with random coin positions	1
coins can be dragged	2
dropped coins end up centered in correct location	1
illegal moves are not allowed	3
game over is indicated correctly	2
general correctness	2
appropriate comments (including JavaDoc)	2
style and formatting	2
submitted correctly	1

NOTE: Code that does not compile will not be accepted! Make sure that your code compiles before submitting it.