Graphic Silver Dollar Game

Assignment 1

Due Monday, January 30, 2012

CSC 062: Spring, 2012

The Problem

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.

1. Read through the CS 062 style guide linked on the course web page under Documentation and Handouts. You will need to follow these guidelines 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've forgotten how to do this, see the instructions on the Documentation page for the course.

   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. There are three files `Coin.java`, `CoinSquare.java` and `GraphicsCoinStrip.java` all of which can be found at `/common/cs/cs062/assignments/assignment1/`.

   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.

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
- `@version` with the 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)

Please include pre and postconditions in the headers of methods as appropriate.

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.

Submitting Your Work

- From within Eclipse, select "Export" from the "File menu.
- Click on the triangle next to "General" in the dialog box. Select "File system" and click next.
- Make sure all the files in the dialog on the right hand side are checked, and then click the "Browse" button next to the "To directory:" entry
- Select "Desktop" from the pulldown menu and click on "Choose".
- This should create a new folder on the desktop. You should now rename that folder by clicking on it and pressing return, and typing in a more descriptive title that identifies you and the assignment you are working on, such as "Bruce-Assignment1." (except replace "Bruce" by your last name!). Note that dashes in names are OK, but don't include spaces or periods.

  You must identify that the submission is for an Assignment, so that we can distinguish your lab submission from your assignment submission.
- Quit Eclipse.
- Now open the "cs062" folder icon on the desktop by double-clicking on it. Within the "cs062" folder you should see a "dropbox" folder.
- Drag the folder you just created into the dropbox folder. When you do this, the computer may warn you that you will not be able to look at this folder. That is fine. Just click "OK".
- You can now drag the folder on your desktop into the trash. Do save the original folder in your workspace in case you need to access your program again. (So only the folder you "exported" should be thrown away.)

If you submit your program and later discover that your submission was flawed, you can submit again. We will grade the latest submission made before the 11:59 p.m. deadline. The computer may not let you submit again unless you change the name of your folder slightly. It does this to prevent another student from accidentally overwriting one of your submissions. Just add something to the folder name (like the word "revised") and the re-submission will work fine.