CSE 11 **START EARLY!**
Spring 2015
Program Assignment #2 (100 points)

Due: 17 April 2015 at 1159pm (2359, Pacific Standard Time)

Exercises are ungraded and are not turned in

Note that in all commands, the "$" indicates the shell prompt. You do not type in the $, only what comes afterwards. You hit "enter" or "return" to execute a command in the shell

Exercise #1: Browse to http://blog.linuxacademy.com/linux/ssh-and-scp-howto-tips-tricks/ to learn about ssh (secure shell) and scp (secure copy). These are very useful for remote access to servers and copying files from one Unix host to another.

Exercise #2: Use the commands "`ls`" and "`ls -l`" (that's an "ell" not a one), What does this do? What's the difference between the two?

PROGRAM #2 : Ricochet

READ THE ENTIRE ASSIGNMENT BEFORE STARTING

You are to write a java program called Countdown using the Objectdraw package introduced in class. When your program is complete, your program will look similar to the following two screen shots.

The program is to simulate a second countdown timer, with a second hand and center circle colored blue, at the center of a drawing canvas. The user inputs the size of canvas (in pixels) and a positive number of seconds to count down. A complete circle is just like an analog watch – 60 seconds. 0 seconds is indicated by straight up (or 12 O'clock on a watch). The second hand rotates clockwise.

When the user inputs a positive countdown, say K, the second hand is started at 12 O'Clock – K seconds. For example if the user inputs 15 seconds, the second hand starts at the 9 O'Clock position. If the user inputs 45 seconds, the second hand starts at the 3 O'Clock position. It is not an error for the user to input more than 60 seconds, this simply that more than one full rotation of the second hand should occur.

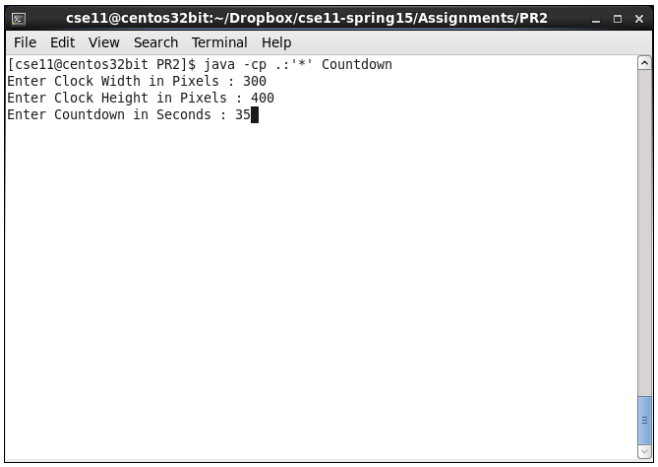
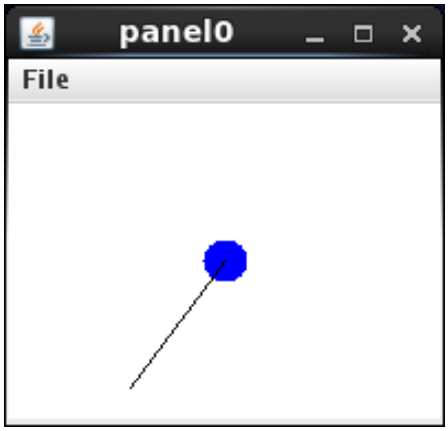


Figure 1 - Sample Text Input



Drawing 2: The countdown secondhand with about 25 seconds remaining

Some Objectdraw Background

You can use any of the graphical example programs presented in lecture as a starting point for setting up your Objectdraw-based final program. You should look at the Objectdraw online documentation, too. Some class from Objectdraw that you will likely find to be very useful for your program are :

- `FilledOval` - use this class to draw and color the center circle
- `AngLine` – use this class to draw the second hand. (You could use `Line`, instead.)

Recall, that the upper-left corner of the drawing canvas has coordinates (0,0) and that positive X coordinates are to the right, and positive Y coordinates are downwards.

It's probably be clear that you need to construct some sort of loop to perform the countdown and move the second hand. We'd like the second hand to move in (approximately) one second intervals (also called a timestep). It may not be obvious to you how make the second hand move. To accomplish this, your loop should being doing "flip book" animation: Draw the second hand at the appropriate place, wait for a timestep, erase the second hand and then redraw it, wait the timestep, erase and redraw, etc. To wait a certain amount of time, T milliseconds, you can use the following code snippet inside of your loop to wait an int T milliseconds

```
try { TimeUnit.MILLISECONDS.sleep(T);}
        catch (InterruptedException e){};
```

You will also need to import the TimeUnit package with

```
import java.util.concurrent.TimeUnit;
```

Since Exceptions have not yet been covered in class, declare and initialize T to an appropriate value and then just use the above code as is.

Program Input

In your `main()` method, you should ask the user for three inputs: the height of clock, the width of the

clock and the number of seconds to count down. Your program should prompt the user in the order listed here

```
Enter clock width in pixels :
Enter clock height in pixels :
Enter countdown in seconds :
```

- clock width and clock height must be positive
- countdown must be greater or equal to 0

The following CONSTANT must be defined in Countdown.java to define the diameter of the center circle. Please put any (other) constants that you define on their own separate line(s). If you change the numbers in the constants, your program should behave differently. In grading your program, we may use different values for this constant and then recompile to verify that your program is actually using this constant. We will only check with reasonable (positive) values of CDIAMETER.

CDIAMETER = 10;

Functionality of Countdown

1. Your name, ID, email must appear in comments exactly the same way they did in Program #1.
2. You should use the entered width and height of the clock as the width and height parameters to the `startController` method. In objectdraw, this specifies how big the window should be.
3. You program should then find the actual width and height by invoking appropriate methods on the `canvas` object. Look at Objectdraw documentation for `DrawingCanvas` to discover the appropriate methods.
4. The center circle must be centered on the drawing canvas, based upon the actual canvas width and height. Centered means the center of the circle is at the center of the of the drawing canvas
5. The center circle should be colored blue.
6. You must calculate the length of the second hand so that it is of fixed length, but would never be drawn beyond the visible canvas. Its length will be determined by either the actual canvas width or actual canvas height.
7. The minimum length of the second hand is 10 pixels greater than the radius of the center circle.
8. The second hand should start at the appropriate point on the clock face and count down seconds. If countdown seconds is greater than 60, the second hand will make more than one full sweep of the clock face.
9. Once the countdown has completed, the secondhand should be in the 12 O'Clock position and all movement should stop. The program should not exit automatically at this point.
10. Your program should check for some forms of bad input. If any of the following are determined, your program should print out the phrase "BAD INPUT" and then exit
    1. a non-positive input height
    2. a non-positive input width

3. a negative height
4. An input height or width that would make it impossible for the secondhand to be its minimum length. (base this error condition on the input height and width not actual height and width)

11. If the actual canvas height and width would cause the minimum second hand length requirement to be violated, set the length to the minimum allowed and proceed. In this case (and only this case), the second hand (and or center circle) could be drawn outside the viewable canvas.
12. Once the countdown is complete, when the user clicks on the canvas, the program should exit. You may use the method call `System.exit(0)` to accomplish this.

Some hints on building Countdown

1) Read the specifications carefully, so that you fully understand what you are trying to accomplish.
2) The `main` method should read the user inputs, store them in appropriately named variables, and then use the `startController` method as described in class
3) The rest of the program can be coded in the `begin()` method. You may define other helper methods if you want to, but it is not required.
4) Once the `begin()` method has completed, Objectdraw will process mouse events. You should define an `onMouseClick()` handler to exit the program when you click on the canvas.
5) Do not attempt to build the entire program at once and then debug the result. Instead, build in stages.
6) A good plan for building this program in stages might be the following: (Be sure to test/verify your program works properly at each stage before moving to the next)
   a) Read the user inputs, print them back out to make certain that you are reading variables correctly
   b) Check that the program exits and prints `BAD INPUT` under the conditions stated in the assignment
   c) Create a centered circle with diameter `CDIAMETER` size in the center of the canvas. Make sure it is centered. Test with a variety of inputs to make certain your calculations are correct.
   d) Color the centered circle correctly.
   e) draw the second hand at its initial setting. Check for various values (e.g. 15, 40, 60, 110 seconds remaining).
   f) Get the second hand to move by drawing, waiting, erasing and then redrawing
   g) Make sure that the second hand stops moving when you've reached 0
   h) code the `OnMouseClick()` handler to exit the program properly

Make Copies of your Program Files as you go along, If you make a big mistake you can go back to the previously working code

Turning in your Program

YOU MUST BE ON THE LAB MACHINES FOR THIS TO WORK. PLEASE VERIFY WELL BEFORE THE DEADLINE THAT YOU CAN TURNIN FILES

You will be using the "bundlePR2" program that will turn in the file

```
Countdown.java
```

No other files will be turned in and and it **must be named exactly as above.** BundlePR2 uses the department's standard turnin program underneath.

To turn-in your program, you must be in the directory that has your source code and then you execute the following

```
$ /home/linux/ieng6/cs11s/public/bin/bundlePR2
```

The output of the turnin should be similar to what you saw in your first programming assignment.

You can turn in your program multiple times. The turnin program will ask you if you want to overwrite a previously-turned in project. **ONLY THE LAST TURNIN IS USED!**

Don't forget to turn in your best version of the assignment.

Frequently asked questions

Can I do this as an applet? No. Countdown must be a Java program.

What if my programs don't compile? Can I get partial credit? No. The bundle program will not allow you to turn in a program that does not compile.

There are a large number of methods to choose from in Objectdraw, can you give us some guidance? See the comments above for appropriate classes. Part of this assignment is to start reading some documentation. In particular look at the methods defined in DrawingCanvas find out how to get the width and height.
The documentation for objectdraw is online:
http://eventfuljava.cs.williams.edu/library/objectdrawJavadocV1.1.2/index.html

My circle can't be exactly centered because of the actual dimension of the canvas and/or CDIAMETER is an odd number, what should I do? Get within one pixel of center, that's the best you can do.

When I use small values (but larger than the required minimums), every looks strange, is that OK. Yes. Just make sure you are testing for bad input as specified above.

Do I have to check for all kinds of crazy inputs? What if a user inputs a floating point number or text when it should an integer? Check the conditions for what we've asked for. We will only input integers. Your program respond properly for good/bad integer inputs. You do not have to handle a user entering non-integer input.

Will you dynamically resize the canvas during grading? No.

Do the input prompts need to look just like the example? Yes. Including the colons. This allows a grading script to properly process your program out.

Can I add extra output to my program? No.

.

Can I add extra graphics items to make it look more like a clock? Yes. .

I need the constant π, what should I use? `Math.PI`.

START EARLY! ASK QUESTIONS!