CS112 - INTRODUCTION TO PROGRAMMING

Programming Project #2 - Due Wednesday, March 2, at the start of class

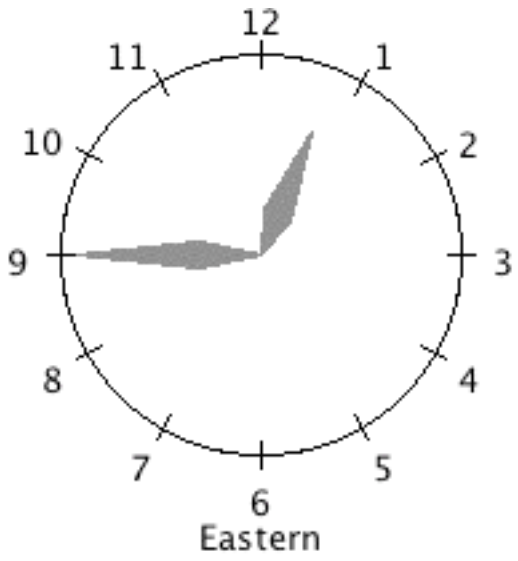
Purposes:

1. To give you further experience with Java arithmetic and the use of the Math class.
2. To give you further experience with drawing in Java using the methods of the Graphics class.

Introduction

This project is based on a variant of problems 5.21 and 5.22 in Wu. As you recall, problem 5.21 was the basis for Lab 5 in this course.

What you will do for this project is to create a Java applet that - as a minimum - displays a nice clock and two input fields labelled "Hours" and "Minutes". When the user enters a time into these fields and presses the "Enter" key on the keyboard, the clock will be set to the new time. Example:



Hours (EST): 12 Minutes: 45

Your program must consist of a minimum of two classes: an applet class and a clock class. The latter will be based on the class you created in Lab 5, but with various improvements, as discussed below. You may enjoy looking at versions of a similar project created by students in previous years, linked off the course home page.

After the projects are completed, your applet will be posted on the department's web server so that fellow students, parents, friends, etc. can see what you've created as well.

Evaluation

Your grade on this project will be based on three criteria:

1. Correct, neat, and visually pleasing operation. (maximum 40-60 points, depending on options chosen)
2. Good methodology, including documentation, use of comments, overall structure, choice of names, and good use of white space to aid readability (indentation and blank lines). (maximum 20 points)
3. A project quiz, to be given on the due date. (maximum 20 points)

A blank project cover sheet is attached and should be stapled to the front of your project submission.

Requirements

Minimal requirements - up to 40 points for correct operation

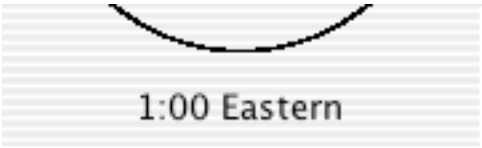
Fulfill the requirements as stated above. Improve the appearance of the clock you created in Lab 5 in some or all of the following ways:

- Add numbers and/or markers around the face of the clock
- Improve the appearance of the clock hands. We'll discuss possibilities for this in class.
- Use color to make the clock more visually appealing.
- Other creative possibilities of your own devising

Credit will be based on the aesthetics of the finished product. It is more beneficial to do what you do well than to do lots of things sloppily. Full credit will require doing all of the above, while demonstrating that you understand the formulas for positioning hands, markers etc. and that you have learned something beyond what we have explicitly covered in class or lab.

Option 1 (Maximum of 5 additional points when added to the minimal requirements)

Each clock should display the time both in digital form and in analog form. The digital time should be displayed before the label, just below the clock. Refer back to the discussion of formatting integers in class to be sure that times are correctly formatted - e.g. 1:03 should be displayed as 1:03, not 1:3 or 01:03. **This must be handled by the clock object - not separately by the applet.** (I.e. the clock object must display the digital time in its `paint()` method.) The following illustrates what this might look like:



Option 2 (Maximum of 5 additional points when added to the minimal requirements - can be done with or without Option 1)

Consider the task of drawing "hash marks" and time labels as in the example on page 1. This could be done by a series of statements like:

```
graphics.drawString("1", calculated X position for 1:00 hour label,
                         calculated Y position for 1:00 hour label);
graphics.drawLine(calculated start X for 1:00, calculated start Y for 1:00,
                  calculated finish X for 1:00, calculated finish Y for 1:00);
```

```
graphics.drawString("2", calculated X position for 2:00 hour label,
                      calculated Y position for 2:00 hour label);
graphics.drawLine(calculated start X for 2:00, calculated start Y for 2:00,
                    calculated finish X for 2:00, calculated finish Y for 2:00);
...
graphics.drawString("12", calculated X position for 12:00 hour label,
                      calculated Y position for 12:00 hour label);
graphics.drawLine(calculated start X for 12:00, calculated start Y for 12:00,
                    calculated finish X for 12:00, calculated finish Y for 12:00);
```

It would be much nicer, however, if this were done using a for loop:

```
for (int hour = 1; hour <= 12; hour ++)
{
      calculate position for hour label and hash marks
      graphics.drawString(""+hour, calculated X position, calculated Y position);
      graphics.drawLine(calculated start X, calculated start Y,
                          calculated finish X, calculated finish Y);
}
```

Where the positions for each hour label and hash mark are calculated based on the hour and the diameter of the clock. Implement both the drawing of 12 hash marks and 12 hour labels this way. (Hint: it is easy to calculate the position for the center of each label. It will take a bit of "tweaking" to get the label position to come out exactly right - especially the centering of the hour labels at 6:00 and 12:00, and the fact that the right hand side of the hour label is close to the clock for 7:00-11:00, while the left hand side of the hour label is close to the clock for 1:00-5:00. See the `Hello` class from Lab 5 for an example of how to center a label) The maximum of 5 points for this option will be given for hour labels and hash marks positioned perfectly symmetrically and elegant code.

Option 3 (Maximum of 10 additional points when added to the minimal requirements - can be done with or without Options 1 and/or 2)

North America encompasses a total of 8 time zones - listed here in order from east to west: Newfoundland, Atlantic, Eastern, Central, Mountain, Pacific, Alaska, and Hawaii. Information on converting times between these can be found at `http://atm.geo.nsf.gov/ieis/time.html` (all but Newfoundland) and `http://www.education.gov.nf.ca/liv_rel_time.htm` (Newfoundland Time). Modify your applet as follows:

- Instead of displaying one clock, display eight clocks - each showing the time in a different time zone. The label below each clock should indicate its time zone.
- One clock (Eastern Standard or the time zone of your home) should be displayed noticeably bigger than the other clocks (e.g. perhaps twice as big). The user will input the time in this zone, and the 8 clocks will each be set to the corresponding time in their zone. (Be sure your labelling of the input fields makes it clear to the user what zone the input is supposed to be in.). You don't need to worry about daylight savings time - after all, it's still winter!

If you do this option, you'll need to take into account issues like the fact that the time 1 hour before 1 is 12, and the fact that Newfoundland time differs from EST by 1 hours 30 minutes. Also, give some time to making your overall screen layout aesthetically pleasing.

Implementation Notes:

1. To start the project, copy the Project2 folder from the common volume to your server volume. This folder contains a BlueJ project and the applet class you will need to use (`Project2.java`). This class can be used essentially as is (with some aesthetic adjustments,

perhaps) unless you do the multiple clocks. You will need to create the `Clock` class and an `html` file to actually run the applet. Of course, you wouldn't forget to add appropriate comments to the `Project2.java` file as well!

2. If you do the multiple clocks option, you will need to add a considerable amount of additional code to `Project2.java` to handle the extra clocks. This will require

   • adding instance variables for each clock
   • changing the comment "Create the clock" to "Create the clocks" and adding code to create, add, and position each clock to this section.
   • arranging to call `setTime()` for each clock when the user enters a new value.
   • possibly changing the `setBounds()` calls for the `hours` and `minutes` labels and `hoursIn` and `minutesIn` text fields to change their position on the overall screen.

3. If you do the multiple clocks option, each clock should be set to show the correct initial time relative to 12:45 in your primary time zone (EST or a different zone you have chosen). (I.e. if your Eastern time zone clock shows 12:45 initially, your Atlantic time zone clock should show 1:45, your Central time zone clock should show 11:45, Mountain 10:45, etc.) When the user updates the time, all clocks should be set to show the correct time relative to the new time the user entered in the primary time zone.

   Hint: This is most easily handled by adding two instance variables (say `deltaHours` and `deltaMinutes`) to each clock, which record the amount of time by which the clock's time zone differs from the primary time zone - e.g. if the primary time zone is Eastern, `deltaHours` for Atlantic would be 1, while `deltaMinutes` would be 0; for Newfoundland, `deltaHours` would be 1 and `deltaMinutes` would be 30. Include this information as two additional parameters to the constructor and specify the correct values for each clock as it is constructed. Use each clock's `setTime()` method to set it to its equivalent to 12:45 Eastern at startup.

4. A demonstration of the *functionality* you are expected to achieve if you do all options is accessible from the course web page. Please note that this example does <u>not</u> use color or other features to produce good aesthetics - it simply behaves correctly. (Maximal credit requires more than this), For maximal credit, your project should be <u>both</u> functional and aesthetic. (But a beautiful program that does not work correctly is not a beautiful thing!)

Turn in the following, neatly stapled in the order listed:

1. Project Coversheet (attached)
2. Documentation including Problem Statement and Design Document. (Note: you did similar documents in Lab 5, but the ones you turn in with the project should reflect the requirements of this project, not the lab.)
3. Printout of the java sources for the classes you created, and of your html file. Note that you are just to turn in a single program, reflecting the highest option you did. However, you will probably find it wise to attempt options successively, and to save a copy of an option before moving on to the next in the case of catastrophic error. Be sure you have deleted unnecessary "boilerplate" code creaetd by BlueJ.

Leave on server:

The BlueJ project folder containing an executable form of your classes. Also, drag the `Project2` folder - **<u>with its name changed to *your username*2</u>** to the CS112 Drop Box, following the procedure that will be demonstrated in class. (Note: you may **<u>not</u>** make any changes to the files in the folder after you have turned the project in! However, you may change the submitted version in the Drop Box prior to the due date by replacing it with a new version.

CS112 - INTRODUCTION TO PROGRAMMING - PROJECT TWO

Option(s) attempted-check all that apply: ❑ 1 ❑ 2 ❑ 3 Author _____

I. Correct Operation / Neat and Aesthetically-Pleasing Output

Points _____

II. Methodology

1.External documentation exhibits clear understanding of the problem and demonstrates good planning for solving it.

Definitely 3 Mostly 2 Partially 1 Not at all 0

2.Prologue comments for files, classes and methods make the purpose of each item clear.

Definitely 3 Mostly 2 Partially 1 Not at all 0

3.Identifiers (class, method, and variable names) clearly describe the item they name and follow OO naming conventions.

Definitely 3 Mostly 2 Partially 1 Not at all 0

4.Features of the Java language and class library are used appropriately and efficiently.

Definitely 3 Mostly 2 Partially 1 Not at all 0

5.Whitespace (indentation and blank lines) follows a consistent convention and makes the overall structure of the program clear by enhancing its readability.

Definitely 3 Mostly 2 Partially 1 Not at all 0

6.Overall methodology results in a program that is easily understandable and obviously correct

Definitely 5 Mostly 3 Partially or not at all 0

Points _____

III. Quiz

Points _____

OVERALL TOTAL (Max 100) Points _____