CS1110 Lab 02. Creating objects, calling their methods, and writing subclasses Fall 2010

Name _____ Netid _____

We ask you to experiment with creating objects, calling their methods, and writing subclasses. You'll also take a look at Java API specifications and generate some Javadoc specifications yourself. At the end of this handout is a list of oft-used methods of class JFrame.

Task 1. Practicing calling methods of objects

(a) In the DrJava Interactions pane, create a JFrame object, storing its name in variable jf1 (which you'll need to declare), and show it. For the creation and storage part, either use the assignment statement

```
jf1= new javax.swing.JFrame();
```

or use the two-statement sequence below, which employs an import statement (Sec. 1.3.2 of the text) —execute

```
import javax.swing.*;
jf1= new JFrame();
```

Now create and show a second JFrame object using a different variable jf2. Check that you can drag both windows around to different places.

(b) Execute calls for the eight methods shown in the table below. The three dots ("...") denote an argument that you have to fill in when you type the call into the Interactions pane. For a function call, write down next to its name below the value that the function call returns. For a procedure call, just put a check next to it. You can call these methods as many times as you want. Experiment.

show();	getWidth()
setSize(..., ...);	getHeight()
setLocation(..., ...);	getX()
setTitle(...);	getY()

Describe BRIEFLY in the space below the syntactic and semantic differences between the four methods (and calls on them) in the left column and the four in the right column. Syntax refers to grammar, or structure; semantics refers to meaning —in programming, how a statement is executed or an expression is evaluated.

Task 2. Positioning JFrames

Reset the Interactions pane with the "Reset" button. Create two new JFrames, assigning their names to variables. Use method calls to make the first window 150 x 220 pixels and the second window 220 x 150 pixels. Which coordinate comes first, the horizontal or the vertical one? (Write your answer below). Check your answer by calling methods getWidth and getHeight of one of the JFrames.

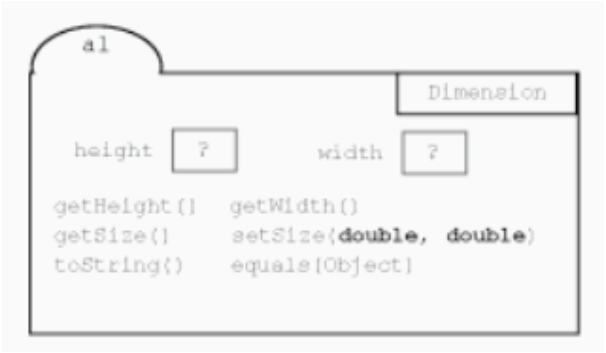
Drag the first window so that it is on the left of the screen and halfway down. Then, write down its x-coordinate and y-coordinate, determining them via method calls.

Task 3. Customizing a JFrame

1. Class Toolkit in package java.awt has a method for obtaining the screen size of your monitor (number of pixels in both dimensions). Type or copy/paste the following into the DrJava Interactions pane:

```
import java.awt.*;
Dimension d= Toolkit.getDefaultToolkit().getScreenSize();
```

To the right is an object of class Dimension (in package java.awt), so you can see what it looks like. Fields height and width are of type **int**. There are methods for obtaining the width and height fields.



Function toString yields a String that describes the object. To see this description, type this in the Interactions pane:

```
d.toString()
```

You should then see the dimensions of your monitor. Copy them here: _____, _____ Also, type (or copy/paste, one at a time) in these expressions into the Interactions pane:

```
d.getHeight()
d.getWidth()
```

2. Type a new class named MaxWindow into the class window (upper right pane) of DrJava. That is, type this:

```
import javax.swing.*;
import java.awt.*;
/** An instance is a JFrame with methods to maximize its dimensions.
*/
public class MaxWindow extends JFrame {

}
```

Save the file as MaxWindow.java, using that exact capitalization in the filename, putting it in a NEW SEPARATE FOLDER. *Every time you start a new Java project, place its files in a new folder*. Now compile MaxWindow.java (press the compile button). Then use the Interactions pane to create and show a MaxWindow. This is just to make sure that your basic class definition is correct.

3. In MaxWindow, add this method (if you want to cut/paste instead, see below first)

```
/** Move the JFrame to the left of the screen and make its width
    the width of the screen */
public void maximizeWidth() {
    setLocation(0, getY());
    Dimension d= Toolkit.getDefaultToolkit().getScreenSize();
    setSize((int) d.getWidth(), getHeight());
}
```

The above version is indented properly, but if you copy and paste you may get errors because of the spaces used to indent. Instead, copy the version given below, paste it into the class definition, select all the text in the pane, and either hit the "tab" button or use DrJava's indent command (in menu Edit) to indent the class properly.

```
/** Move the JFrame to the left of the screen and make its width
the width of the screen */
public void maximizeWidth() {
setLocation(0, getY());
Dimension d= Toolkit.getDefaultToolkit().getScreenSize();
setSize((int) d.getWidth(), getHeight());
}
```

4. Study method maximizeWidth, so that you know what it does and how it does it. Then, compile the class, switch to the Interactions pane, and create and show an object of class MaxWindow. Drag the window to the right a bit and then call its method maximizeWidth to see whether it works properly.

5. Add to class MaxWindow a procedure maximizeHeight with the specification shown below. In writing the body of this procedure, use the body of maximizeWidth to get some ideas. This will force you to understand what maximizeWidth does. Then compile and test the procedure.

/** Move the JFrame to the top of the screen and make its height the height of the screen */
public void maximizeHeight()

6. Add to class MaxWindow a procedure with the spec shown below. Its body should call methods maximizeHeight and MaximizeWidth. Then compile and test it.

/** Move the JFrame to the top-left of the screen and make it fill the whole screen */
public void maximize()

Task 4. Resizing a JFrame.

Create a new JFrame, storing its name in a variable jf, and show it. Try resizing the JFrame with your mouse to make it bigger. Call function isResizable() of jf to see whether JFrame jf is resizable, and put the answer here (yes or no):

Now execute the procedure call jf.setResizable(false);. Try resizing jf with your mouse. Is it resizeable?

Call function isResizable() in folder jf and put the value it returns here:

Make jf resizable again.

Task 5. Finding the API specifications for method setTitle of class JFrame. This task gives you practice with the general way in which one browses the API specifications.

(a) Find the root of the API specs from the "Links" subpage of the website for CS1110. Open the API specifications for class JFrame —or, simply copy and paste this URL into your browser:

http://download.oracle.com/javase/6/docs/api/javax/swing/JFrame.html

(b) Just read a bit to familiarize yourself with the page —spend a minute looking at it.

(c) Click the mouse near the top of the window, so that the cursor appears near the top. Then search for "setTitle" —do this by typing control-F (if you are on a PC), typing setTitle into the search window, and hitting the return key. The page should scroll down to a part labeled "Methods inherited from class java.awt.Frame", and "setTitle" will be highlighted.

(d) Click on the highlighted term "setTitle". A new page for class Frame will open in the browser window. Read about method setTitle. Then scroll to the top of the page and spend 1/2 minute reading about class Frame.

Task 6. Generate Javadoc specifications.

Specifications formatted like the webpages you have just examined can be created automatically for classes that you yourself write.

Click the Javadoc button in DrJava. (You'll be asked where to save the result.) You should see a browser window open with a description of class MaxWindow. Importantly, the comments you put in your class definition that are surrounded by "/** .../*" appear in the generated javadoc: you should see your class specification "An instance is a JFrame ...", and, down in the Method Summary and Method Detail sections, the specifications for the methods you wrote.

Check to see that this is indeed the case.

Task 7. Show your lab instructor or a consultant your file MaxWindow.java and this sheet.

Here (possibly continuing on to the back of this page) are methods of class JFrame that we tend to use often:

window.show();	Show window window.
window.hide();	Hide window window.
window.getHeight()	= height of window window, in pixels
window.getWidth()	= width of window window, in pixels
window.setSize(w,h);	Set width and height of window window to w and h
window.getX()	= x-coordinate of top left corner of window window
window.getY()	= y-coordinate of top left corner of window window
window.setLocation(x,y);	Set x- and y-coordinates of top left corner of window to x, y
window.getTitle()	= the title of window window (in the title bar)
window.setTitle(s);	Set the title of window window to s (a String).
window.isResizable()	= “window window can be resized by dragging it”
window.setResizable(b);	If b is true (false) make window resizable (not resizable).