Assignment #6

Due: Tuesday, May 21 by class-time

The current assignment is in Java, and the idea is to write an application that gives us some insight into what's happening behind the scenes in a running Swing application. This is a useful thing to understand a UI toolkit, in order to know how event-handling is done, and what attributes the widgets have.

To see what kinds of attributes the typical Component has, take a look at the Component page in the class API documents. In the Method Summary, you will see many symmetric pairs of getter/setter methods, such as:

- getBackground / setBackground
- getName / setName
- getFont / setFont
- getVisible / setVisible
- etc..

There is also a method called:

- getParent

A good way to easily identify the attributes that a widget has is to implement a simple ComponentInspector class. This class can be instantiated, then “attached” to a particular component. The ComponentInspector gives a visual display of that component’s attributes, and lets the user change certain attributes. The most straightforward way to set up the ComponentInspector is to lay out a set of JLabel/JTextField pairs, with each pair corresponding to a single attribute of the selected component. The JLabel gives the name of the attribute, and the JTextField shows the current value, and is user-editable if appropriate. You can use a keyAdapter to listen for the “return” key in the JTextField, and then use whatever the user has typed into the text field to set a new value for the particular attribute. For attributes which take objects other than strings (i.e. Color, Font, etc..), your code should make a best-effort try to instantiate the appropriate type of object from the string, and just throw an exception if something fails.

This information display/edit behavior will require you to make use of the getter and setter methods mentioned above. You should choose 5-10 component attributes to display in your ComponentInspector (please include the Component’s parent as one of these, although this will not be user-editable).

Your ComponentInspector should have a method as follows:

- void attach(Component c)

This method is how you can attach the ComponentInspector to a new Component. When executed, this method will store the reference to the new Component, and update all of the JTextField’s to show the new values.

The other part of this assignment is to build a Swing application with some of the more complex Swing components we have discussed, such as menus, toolbars, or dialogs. Attach ButtonClick listeners to all of the Components, and each time a Component is clicked on, attach your ComponentInspector to the clicked-on Component. You'll need to find a way to spy on the events generated, because we also want your ComponentInspector to have a JTextArea which shows the most recent event (getting the textual representation of an event can be as simple as executing its “toString” method.) It may help that most components can have listeners attached, or to look at the "Glass Pane" principle that David talked about.

Submission: As you did last time, make a directory in the class submission directory with your username (or your usernames, hyphenated together if you are working in a group). The submission directory for this assignment is at:

/afs/ir/class/cs377a/submit/assn6/