Lab Exercise 4

Objective: Write a complete application from scratch.

Required files:

None Ð you write everything from scratch!

Tasks:

Write a painting program which lets the user draw in a scrollable view. The user should have a choice of brush shape (round or square) and brush size (from 1 to 10 pixels across). The user should also be able to select from a variety of paint colors.

The brush shape should be controlled by a radio button pair. Brush size should be controlled by a slider, with nearby text to tell the user what the slider's current value is. Paint color can be controlled by a slider or by radio buttons; in the solution provided we offer a choice of four colors, controlled by four radio buttons. There should also be a ªClearº button.

When the user pushes down the left mouse button, your program should draw a ªdrop of paintº of the correct shape, size, and color. Your program should then start catching mouse-dragged events, drawing drops of paint each time, until the user releases the mouse button.

Since your ªcanvasº is required to scroll, you will have to figure out a way to repaint the DocView as portions of it get scrolled into sight. The easiest way is to have a Bitmap object which contains a copy of the entire ScrollView. When the user draws, draw into the Bitmap as well as into the ScrollView. When the user scrolls, the DocView's drawSelf:: method should composite the appropriate part of the Bitmap into the DocView.

Hints:

1) Start by laying out your window and controls using the Interface Builder. Most of your window will be occupied by the ªcanvas,º which will be a custom view.

2) The ªcanvasº custom view should be a subclass of the ScrollView class. Its newFrame: method should instantiate a fairly large DocView, which should be a subclass of View. Take a little time to think about what functionality should be performed by each object. You will need separate .m and .h files for each of these two subclasses.

3) Your DocView will have to override the mouseDown: method. Your new method should draw paint and then go into a loop, grabbing mouse-dragged and mouse-up events off the event queue. Mouse-dragged events cause more paint to be drawn; mouse-up breaks the loop. Read about the eventMask, addToEventMask:, and setEventMask methods of the Window class, and the getNextEvent: method of the Application class, in the Class Descriptions section of your documentation.

4) A good way to get good performance is to have a little bitmap which represents the image of a single drop of paint. It gets updated whenever the user changes the brush size or shape, or the paint

color. It gets composited to the DocView when it's time to draw one drop. Think about the most appropriate compositing operation.

5) The various controls should affect the DocView, but all you can connect them to with the Interface Builder is the ScrollView. The easiest workaround is for the ScrollView to pass control information on to the DocView. For example, the ScrollView's method to change the brush shape (invoked when the user punches a radio button) should invoke a method in the DocView, passing in the new brush shape as a parameter.

6) Many of the questions you will have are answered in your manuals. This is a good opportunity to get familiar with the documentation - especially the Class Descriptions.