6.893 User Interface Design & Implementation
Fall 2003

Quiz 2

This quiz is closed book, closed notes. You have 80 minutes to complete it.

Your name: _____

1. (3 points) Which of the following problems is most likely to be found by a user test of a horizontal computer prototype?
   a. Response time is too slow.
   b. One dialog box uses a different font than another dialog box.
   c. A dialog box covers up information that the user needs from the main window.
   d. Color choices are bad for blue-yellow color-blind users.

2. (4 points) Name the two main categories of computer prototyping tools, and give one advantage that each category has over the other.

   Storyboard: you can draw anything
   Form builder: provides standard widgets, which actually work

3. (3 points) Which of the following is an advantage of the model-view-controller pattern?
   a. Models can be put inside other models.
   b. Multiple views can display the same model.
   c. Views and controllers are often tightly coupled.
   d. The controller separates the model from the view, so they don't have to know about each other.

4. (3 points) Which of the following is the least related to the view hierarchy?
    a. Input handling
    b. Clipping
    c. Models
    d. Automatic layout

5. (4 points) Give 2 reasons why the view and the controller are often tightly coupled.

    Controllers often need to display output (e.g. menus);
    Some responsibilities span both view and controller (e.g. selection);
    Affordances for control need to be displayed by view;
    Feedback for control needs to be displayed by view.

6. (4 points) List 4 kinds of translated input events.

    Mouse click, double click, drag, key typed, mouse entered or exited, keyboard focus gained or lost.

7. (4 points) The observer pattern is typically used in MVC as a way for the controller to receive input events. How else is the observer pattern used in MVC?

    Views use the observer pattern to listen for changes in the model.

8. (4 points) Louis Reasoner wants to use mouse triple-clicks in his application. He decides to measure the time between clicks by checking the system clock with System.currentTimeMillis(). What's wrong with this approach?

    The mouse events may sit in the queue for varying amounts of time, so the time when the events are actually handled may not be related to the time the user actually clicked the mouse button. Need to use timestamp on events instead.

9. (4 points) Alyssa Hacker is designing a checkers game in which the user makes moves by dragging checkers around the board with the mouse. She's wrestling with a strange bug: if the user is dragging a checker and moves the mouse quickly, the checker sometimes stops following the mouse, even though the user hasn't released the mouse button. The checker freezes in place until the user moves the mouse pointer back over it, at which point it starts moving with the mouse again. (You don't know what UI toolkit she's using, but it isn't Swing.)

   From this behavior, describe:
   a. the output model she's using to represent the checkers;
   b. how she's handling the mouse input for checker dragging;
   c. what's causing the bug, and how it might be fixed.

   Checkers are implemented as components; each checker is handling the mouse move events itself (as opposed to the board handling the events); when the mouse moves so fast that it jumps out of a checker, the system stops dispatching move events to the checker, since the pointer is no longer inside it. Several possible fixes: (1) handle the move events at the board level, (2) use mouse capture to ensure all mouse moves go to the checker during a drag.

   Some people suggested a third fix: turn off mouse move coalescing so that every change in mouse position is seen by the component. This fix is undesirable if the application can't keep up with all the events, since the checker will lag behind the mouse.

10. (4 points) Give one advantage and one disadvantage of using components for output, as compared to strokes.

    Advantages: automatic layout; automatic redraw; input handling (components can receive events, strokes can't).

    Disadvantages: components are heavyweight objects

11. (4 points) Louis Reasoner is writing a Swing widget. He figures he can improve its responsiveness by saving the Graphics object passed to paint(), so that he can change the screen immediately whenever the model changes. Here's some of his code:

```
public class LouisWidget extends JComponent {
    Graphics savedGraphics;
    ...

    void paint (Graphics g) {
        savedGraphics = g; // save Graphics object for later use

        g.drawLine (...);  // draw widget as usual
        ...
    }

    void modelChanged () {
        if (savedGraphics != null) {
           // draw changes right away
           savedGraphics.drawLine (...);
           ...
        }
    }
}
```

Although Ben Bitdiddle thinks Louis's optimization is unnecessary, he's more concerned that Louis has implemented it incorrectly.

What is it about the Graphics object – and analogous objects in other UI toolkits – that makes it a bad idea to save it like this?

Several reasons: (1) Graphics has mutable graphics context state (e.g. font, color, coordinate system, clipping) which may be changed when another component is drawn. (2) The Graphics object passed to paint() doesn't always draw on the screen – it may draw on a printer driver or a temporary memory buffer. LouisWidget can't tell which kind of Graphics object it's saving.

12. (3 points) In the HSV color model, what color is the fully **unsaturated** counterpart of pure red (i.e., identical on all components except S = 0)?
    a. Black
    b. White
    c. Pink
    d. Green

13. (3 points) RGB and HSV color models are both three-dimensional – i.e., each color has three components. RGB is represented by a cube, logically enough. But HSV isn't. How is HSV represented, and why?

    HSV is a cone, for two reasons: (1) hue is perceptually unordered, so it is represented by angle, another unordered variable; (2) value is perceptually dissociative, making hue and saturation harder to perceive as value decreases. Reason 1 alone might make the model a cylinder; reasons 1 & 2 together argue for a cone.

14. (4 points) Give 2 reasons to use your platform's native button widget rather than implementing your own -- 1 reason related to software engineering, and 1 reason related to usability.

    Reuse and external consistency.

15. (3 points) List 3 kinds of user tests.

    Formative evaluation, field study, controlled experiment.

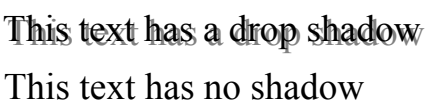
16. (3 points) The double-buffering technique is used for:
    a. Coalescing mouse movements
    b. Preventing screen flicker
    c. Capturing the mouse
    d. Transforming coordinate systems

17. (4 points) In this problem, you will implement an output effect called a **drop shadow**. A drop shadow makes it look like a screen object is casting a shadow by drawing an identical copy of the object shifted slightly in both x and y directions. Here are some examples of drop shadows:

Create a drawing surface ShadowGraphics that automatically draws a drop shadow for every drawn stroke. ShadowGraphics is a wrapper around a Graphics object that does the actual drawing. Here's an example of using ShadowGraphics:

```
void paint (Graphics g) {
    ShadowGraphics sg = new ShadowGraphics (g);
    sg.drawText (0, 0, "This text has a drop shadow");
    g.drawText (0, 50, "This text has no shadow");
}
```

Here's what might appear on the screen for this code:



For the purposes of this problem, a drop shadow is gray and shifted 4 pixels in the positive x and y direction. We'll only implement the methods in Graphics that concern drawing text. You don't need to implement lines, rectangles, etc.

Fill in the skeleton of ShadowGraphics on the next page so that it automatically draws a drop shadow for every drawText() drawing call.

```
class ShadowGraphics implements Graphics {
    Graphics g;



    ShadowGraphics (Graphics _g) {
        g = _g;



    }

    void drawText(int x, int y, String text) {

      // draw drop shadow first, saving & restoring color
      Color c = g.getColor ();
      g.setColor (Color.GRAY);
      g.drawText (x+4, y+4, text);
      g.setColor (c);

      // draw real text on top
      g.drawText (x, y, text);

    }

    void setColor (Color color) {

      g.setColor (color);

    }

    Color getColor () {

      return g.getColor ();

    }

    void setFont (Font font) {

      g.setFont (font);

    }

    Font getFont () {

      return g.getFont ();

    }
    ... // other methods of Graphics
}
```

18. (4 points) Ben Bitdiddle proposes implementing drop shadows for the component model: a container class ShadowContainer that draws drop shadows automatically for all the components inside it. Describe how ShadowContainer could be implemented using ShadowGraphics.

    When ShadowContainer calls paint() on its children, it should pass a ShadowGraphics object wrapped around its own Graphics object.

Suppose you're a facilitator for a user test of a computer prototype. What's wrong with each of the following situations, and what should you do to fix or prevent the problem?

19. (4 points) When the user starts the first task, you discover that the prototype isn't broad enough to cover it.

    Don't waste the user's time. Pilot-test all tasks and materials before the first user comes in.

20. (4 points) During the test, the room is totally quiet.

    The user isn't thinking aloud. Coach the user to think aloud.

21. (4 points) After the test, the user says, "I really felt like an idiot making all those mistakes. I hope you weren't recording me with that video camera."

    User wasn't comfortable and wasn't fully informed. Before the test, tell the user that any mistakes they make are not their fault, but the system's; and tell them whether or not they are being videotaped.

You're trying to use a drawing program on your workstation, but Ben Bitdiddle is logged on to your computer running a huge job in the background, taking up 99% of the CPU. You notice it most when you try to draw a freehand scribble with the mouse.

22. (4 points) If mouse moves are **coalesced** in the event queue, what effect do you see when you try to draw a freehand stroke?

    Stroke is jagged rather than smooth.

23. (4 points) If mouse moves are **not coalesced** in the event queue, what effect do you see when you try to draw a freehand stroke?

    Stroke lags behind the mouse pointer, but follows the mouse path exactly.

24. (3 points) Which of the following is **not** true about the UI toolkit described in the paper "Glyphs: Flyweight Objects for User Interfaces"?
    a. Individual text characters are represented as components, rather than strokes.
    b. Glyphs support automatic layout.
    c. A glyph object must store its screen position.
    d. Glyphs are components (as opposed to strokes or pixels).

Ben Bitdiddle is designing an experiment. He wants to compare handwriting recognition with keyboard text entry.

25. (3 points) The experiment will measure text entry speed and error rate. These measurements are:
    a. independent variables
    b. dependent variables
    c. uncontrolled variables

26. (3 points) The experiment uses a between-subjects design. This decision threatens:
    a. internal validity
    b. external validity
    c. reliability

27. (3 points) Handwriting recognition will be tested on a Tablet PC, while keyboard entry will be tested on a desktop PC. This decision threatens:
    a. internal validity
    b. external validity
    c. reliability

Using different screens and different CPUs for the two interface conditions threatens internal validity: you don't know whether observed differences are due to the input interface or to the other parts of the hardware.

28. (3 points) For each trial of the experiment, users will enter the letters of the alphabet in order. This decision threatens:
    a. internal validity
    b. external validity
    c. reliability

The alphabet isn't a realistic simulation of text input: e.g., the letter frequencies are wrong.