Final Examination

Instructions Carefully read the following questions and provide a reasonably brief answer for each one. Ideally use a word processor to prepare your answers. Alternately you may write your answers, but they must be legible. Please do not include any unnecessary explanations in your answers; focus entirely on the questions asked. Ancillary responses will not increase the value of an answer.

In preparing your answers to these questions rely primarily on the textbook and the materials presented in class (and on the website). You may, if you wish, utilize additional resources, but do not collaborate with anyone else on your responses, whether they are members of the class or not.

Submit your answers to these questions in hard copy (printed or written form, no electronic submissions) no later than noon on Friday, May 4, 2007. Placing them in my mailbox is sufficient, but you may also give them to the departmental secretary if you are concerned about the safety of the delivery method.

Please make certain you identify yourself and the class appropriately on your submission, and staple your pages together firmly so they will not become separated (it’s a good idea to put your name on each page of your submission). You must also attach and sign a declaration that appears at the end of your responses. It must state “I did not accept, receive, or solicit any assistance in the preparation of these responses to the examination.” Responses submitted without this declaration will not be evaluated.

All questions have the same weight.

1. It is sometimes difficult for users new to GUI programming to understand the sequence of events involved in marking (drawing, writing, displaying, etc.) on the screen. After all, in a typical non-GUI environment you just say “print” and printed output appears! Describe, in general terms, the approach used to marking on the display in a GUI application. In particular, what is meant by “destroying” or “invalidating” part or all of the display?

2. The phrase “look and feel” was used to describe various characteristics. First, give a brief definition of look and feel, making certain the terms are differentiated. Then indicate to what they apply. For example, is it just a selected set of widgets, a particular type of application, or all applications? Finally, indicate why competing companies might feel justified in filing lawsuits related to look and feel.

3. What problem do “struts and springs” address? What makes this approach better (or worse) that writing explicit code (e.g. in C++) to solve the problem?

4. Numerous types of user inputs were considered, with most of these having predefined events or messages provided by the system (e.g. mouse button up, mouse move, keyboard entry). Additional events or messages can be defined by the user in most systems. For what reason are such events provided? Give an example or two that illustrates how such events might be used.

5. The phrase “event dispatching” was used to describe the mechanism in a system that is used to determine how events (messages) are “routed” to the appropriate window in an application. Briefly discuss the difference between “bottom up” approaches and “top down” approaches, making it clear what these phrases mean. Then address the issue of dispatching keyboard events, and indicate if the “bottom up” and “top down” phrases make sense for such events. Finally, indicate how user-defined events (as mentioned in question 4) can be dispatched to windows.
6. The Model-View-Controller (MVC) architecture was continually employed to discuss the architecture of GUI applications. Briefly describe this architecture, providing a clear distinction between the functions of each component. Then indicate the value of object-oriented programming approaches when using the MVC architecture, if there are any. Finally, to contrast with the object-oriented approach, indicate what major differences in implementation would be necessary if the object-oriented approach was not used.

7. The Apple Macintosh and Microsoft Windows were repeatedly used as example systems. The graphical user interfaces in these systems are designed for use by a single user at a time (they traditionally have a single keyboard mouse, and display – although multiple displays can be employed, they are still under the control of the single user). This single-user design has effectively dictated the hardware design of many computer systems (which is why they’re called “personal” computers). If we ignore the hardware concerns (e.g. attaching multiple keyboards, mice, and displays to a computer), identify and briefly describe any issues that you feel might be relevant to providing support for multiple GUI users on the same computer system at the same time.

8. What is the usual mechanism used in Microsoft Windows used to make resources (e.g. widget properties) available at the time an application is executed? How is this accomplished in the Macintosh, and in the X window system? In which of those systems do you expect a software tool other than a text editor to be required to modify a resource? Why?

9. Recall that the X window system identifies clients and servers, which may be on separate computer systems connected by a network. Suppose an X client is running on computer “A” and an X server is running on computer “B” (which happens to use Microsoft Windows). Is it possible to copy an object from a window on computer “B” being managed by a Microsoft Windows application to a window associated with the X client running on computer “A”? If not, briefly explain why it isn’t possible. If it is possible, briefly describe the actions taken by the software.

10. Consider the use of the equation editor provided with Microsoft Office in the Word application. Using this tool, users may include graphical representations of equations in their documents. Discuss how the equation is actually stored in the file that contains the document. Since Microsoft Windows can run on different types of systems, using different types of processors, what will happen if the document – with the equation – is manipulated on a system significantly different from that on which it was created?