Do not forget to write the exam sheet ID here (NOT student ID):________________

EXAM Programming of Interactive Systems

21/12/2012—Duration: 2 hours

Authorized material: any document in paper form.
Read the **entire** announcement. Be **clear**, **concise** and **precise**.

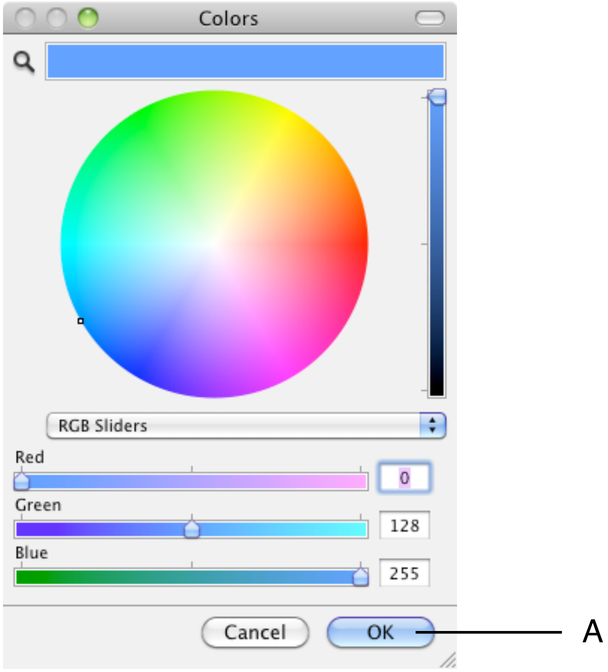
A. Toolkit (3 points)

Figure 1 shows a dialog window for choosing a color in different ways.

1. Identify the components (widgets) you can see here (you can rely on the names used in the Java Swing toolkit).

Note: Label the components directly on Figure 1 with a unique letter and a line pointing to it, as we did below with the “OK” button (letter A), and reference them with that letter on your exam sheet. Hand this page in at the end.

2. Identify the event listeners for the components that need one and specify the most precise event they are listening for (even if you do not know the precise name of the event listener, explain the type of events it listens for).
3. For each event listener, explain briefly what action they perform (in plain english or pseudocode). We suppose that you can access the values of all components on this dialog window.



RGB Sliders

Red 0

Green 128

Blue 255

Figure 1—Color chooser

B. Movement (4 points)

Figure 2 shows a new technique for contextual menus. They are invoked when the user clicks with the right mouse button. The items of the menu are round targets (disks) of different sizes surrounding the cursor. To select an item, the user moves the cursor inside one target and clicks on it with the left mouse button. A click outside any target dismisses the menu.

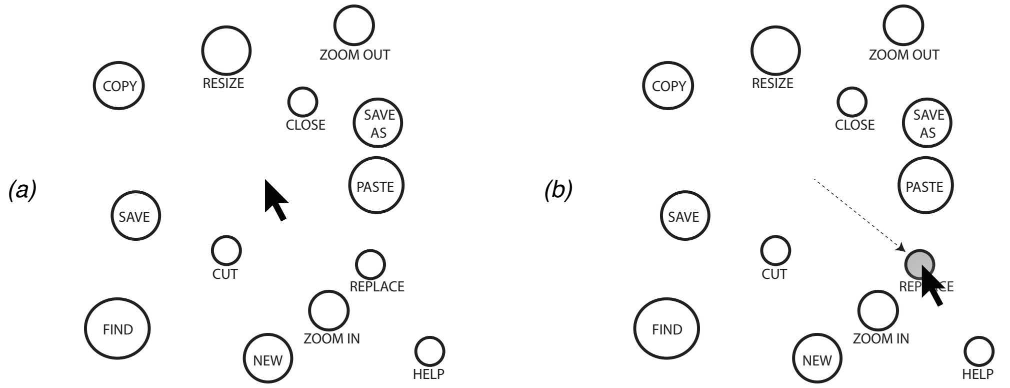


Figure 2—Disk-based contextual menu for a text editor.
(a) Menu invoked. (b) Cursor moved to select item "Replace" by clicking on it.

1. Which item is the fastest to select? Which item is the slowest? Explain your answers.
2. What are the advantages and disadvantages of this technique compared to traditional contextual menus in the form of a list appearing next to the cursor?
3. Imagine that you can define the structure of the menu (position and size of targets). Design your own structure. Justify your choice based on your personal needs and what you know about human capabilities.

Figure 3 presents another contextual menu. The items are lines that can be selected by dragging the cursor across them while holding the right mouse button down. If the right mouse button is released and no crossing occurred, the menu is dismissed.

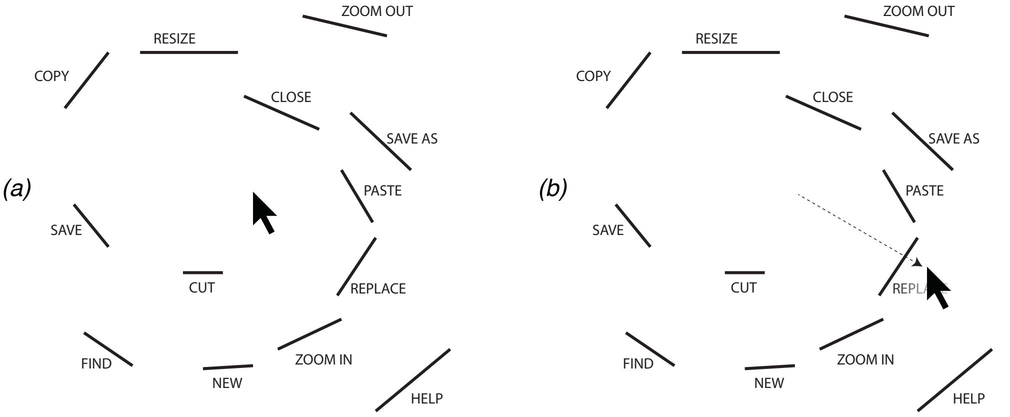


Figure 3—Crossing-based contextual menu for a text editor.
(a) Menu invoked. (b) Cursor dragged to trigger command "Replace" by crossing it.

4. What are the advantages and disadvantages of this menu to the one in Figure 2? Is selection faster or slower?

C. State machines (4 points)

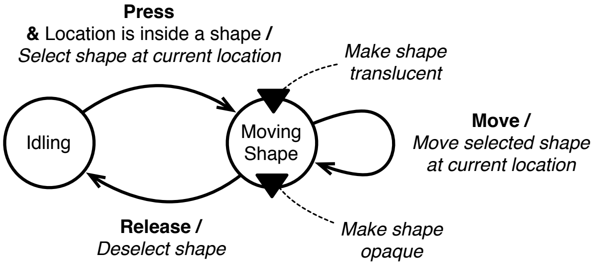

Press
& Location is inside a shape **/**
Select shape at current location
Make shape translucent
Idling
Moving Shape
Move /
Move selected shape at current location
Release /
Deselect shape
Make shape opaque

Figure 4—Example of a state machine describing the dragging of a shape.

In this exercise, we ask you to describe state machines graphically. The actions should be described in pseudocode. Variables and functions can be introduced as long as their meaning is clearly defined. Figure 4 shows an example of graphical representation of a state machine. Circles represent **states** and arrows represent **transitions** between them. Transitions are fired when an **event** (in bold) occurs and if an optional condition, called **guard** (after the "&"), is fulfilled. **Actions** can be associated to transitions (after the "/") or when entering or leaving a state (black triangles).

You can assume the existence of a list of all objects `shapeList[]` and the functions:

- `boolean inside(x, y, shape)` that returns true if point located at (`x`, `y`) is inside object `shape`.
- `boolean crosses(x1, y1, x2, y2, shape)` that returns true if the line segment defined by the points (`x1`, `y1`) and (`x2`, `y2`) has crossed `shape`.

1. Propose a state machine describing the interaction with the menu in Figure 2.
2. Propose a state machine describing the interaction with the menu in Figure 3.
3. Imagine you have access to **EnterOnShape** and **LeaveOnShape** events for simple Rectangle shapes, but *not* Line shapes (Figure 3). Describe a state machine that detects crossing of Line shapes (Figure 3) using these events instead of the `crosses` function. Assume you know the two end points that define all your Line shapes.

D. Design, state machines and evaluation (5 points)

We now want you to design extensions to the techniques seen in exercise B, describe them using state machines, and evaluate them using different criteria.

Note: A simple key modifier solution will not give you full points.

1. For the menu in Figure 2:
   1.1. How would you extend the menu to activate multiple commands during a single invocation? Describe it textually and/or with sketches.
   1.2. Draw the corresponding state machine (you can reference parts from exercise C).
   1.3. Evaluate your design based on any laws of human movement that apply to it (Fitts, Steering law, etc.) and the usability criterion of "visibility".
2. Apply questions 1.1, 1.2 and 1.3 to the menu in Figure 3.

E. Drag-and-drop (4 points)

*In this exercise, limit your answers to **5 lines** per question.*

We have a simple graphical vector application that allows to create, resize and delete shapes. What is missing is the ability to drag shapes inside the document window (Figure 5) or from one document window to another (Figure 6).

When dragging a shape inside a document window, if the cursor reaches a border of the window, the viewport should start scrolling in the appropriate direction after a short delay, even if the cursor stops moving. The ghost of the dragged shape should follow the cursor.

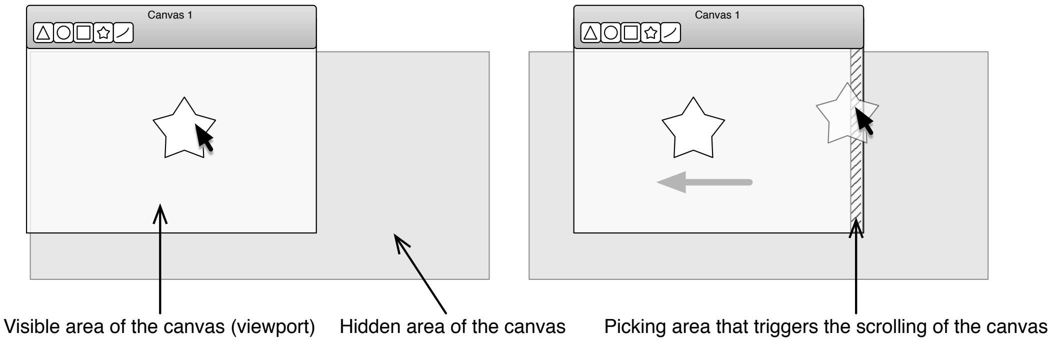

Canvas 1
Canvas 1
Visible area of the canvas (viewport)
Hidden area of the canvas
Picking area that triggers the scrolling of the canvas

Figure 5—Drag and drop of a shape inside a document with automatic scrolling if the cursor dwells for a short delay near the window border.

When dragging a shape between two document windows, the ghost of the dragged shape should follow the cursor anywhere on the screen, even when outside of the window.

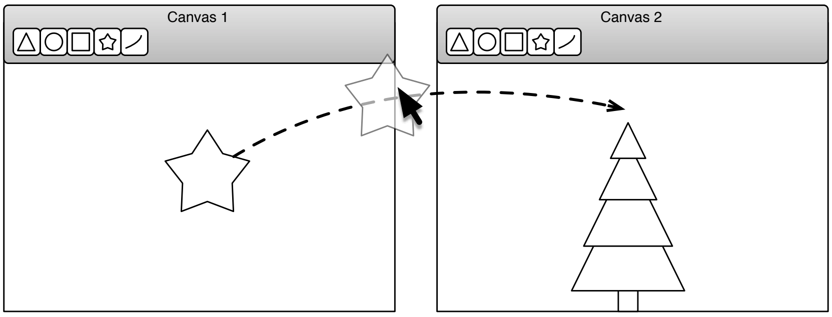


Figure 6—Drag and drop of a shape from one document to another.

1. How would you ensure programmatically the automatic scrolling interaction?
2. If the shape needs to be dragged to a location on the canvas that needs a lot of scrolling, this can be time consuming. Thus, allowing to control the scrolling speed could accelerate the interaction. How would you allow to control the scrolling speed? How would you implement this?
3. While dragging, it happens that some mouse movement events are lost. Give an effective way to handle programmatically the dragging of an object with the mouse cursor by detailing the actions (variables stored and computations in pseudocode) of the transitions **Press**, **Move** and **Release** (Figure 4).
4. How would you make sure that the shape being dragged follows the cursor anywhere on the screen (even outside of the document window)?