1 Introduction

Write each program using either Code::Blocks on Windows or Linux, or XCode on the Mac, zip all the projects into one zip file and upload to the assignment page of the MyClasses site for this class. In these exercises you need to update the DoxyGen documentation. Fully document all new data members and methods to any of the classes, and update the documentation for anything that has changed. Of course, include your name as an author on any files that you edited. There are Windows executables for each program for you to run and experiment with.

2 Exercises

1. The following exercise is an update of the Keyboard State Processing program example. Read through the Box class carefully and make sure that you understand all of the code. Make the following changes to the project.
    (a) Make the title bar display "Homework #3 — Program #1".
    (b) In the keyboard state processing function of the UI class, remove all the features in the alt and shift modifier sections. Also remove the code from the alt modifier in the keyPressed function.
    (c) Add in the following if either alt key is pressed, B will turn the box solid blue, G will turn the box solid green, W will turn the box solid white, and M will reset the box vertices to the red/green/blue/white default vertex colors.
2. Update the previous program as follows.
    (a) Make the title bar display "Homework #3 — Program #2".
    (b) Add in the mouse functionality to highlight the box in red when the mouse is over the box and revert to its original colors when the mouse is not over the box. Here I would suggest to create a method inside the box class that will return a boolean of true if the mouse point is inside the box and false if the mouse point is not inside the box. You can then use this function to reset the color of the box.
    (c) Add in the mouse functionality to do a click and drag movement of the box. So if the mouse is over the box then a left click and drag will move the box along with the mouse. This is to be done smoothly with no "snapping" to the center of the box. So if the user has the mouse in the upper left corner of the box, does a click and drag, the mouse stays in the upper left corner of the box, it does not change the position to the center of the box.
3. Update the previous program as follows.
    (a) Make the title bar display "Homework #3 — Program #3".
    (b) The program will start with a blank screen.
    (c) If the control is down and the user right clicks the mouse on a place where there is no box then the program will place a box on the screen with center at the click position and a random width and height between 0.1 and 0.3 in length. The color of the box is to be a single color chosen at random from all possible colors.

        If the control is down and the user right clicks the mouse on an existing box then the box is removed. You may restrict the total number of boxes on the screen to 100.
    (d) If the mouse hovers over a box the box should be highlighted in red and then return to its original color when the mouse leaves the box.

(e) If the user left clicks a box with the control key down, the current box will toggle a selected mode. Selected boxes will be visible to the user by a yellow outline to the box. This will take some work. One way to do this is to create another VAO to handle the outline, and load this separately to the graphics card. The outline VAO would have the four vertices of the box and use a line loop to graph them. When displaying the box you would draw the filled box and if the box was selected you would then draw the outline over it.

(f) Control+A will select all of the boxes and Control+Q will deselect all of the boxes.

(g) If the control key is down and the user clicks outside of any of the boxes then all boxes are deselected.

(h) If the mouse is over the box then a left click and drag will move the box along with the mouse. This is to be done smoothly with no "snapping" to the center of the box. So if the user has the mouse in the upper left corner of the box, does a click and drag, the mouse stays in the upper left corner of the box, it does not change the position to the center of the box.

(i) The following options will apply to selected boxes only, but to all the selected boxes together.

- If no modifier keys are pressed:
  - Left: Moves the box to the left.
  - Right: Moves the box to the right.
  - Up: Moves the box up.
  - Down: Moves the box down.
- If the control key is down:
  - Left: Decreases the width of the box.
  - Right: Increases the width of the box.
  - Up: Decreases the height of the box.
  - Down: Increases the height of the box.
- If the alt key is down:
  - Up: Increases the height and width of the box.
  - Down: Decreases the height and width of the box.

(j) If the Shift key is down, and the mouse is over a selected box, then a left click and drag will move all of the selected boxes the box along with the mouse, but only the selected boxes. This is to be done smoothly with no "snapping" to the center of the box.

(k) Other program features:

- If no modifier keys are pressed:
  - Escape: Ends the program.
  - M: Toggles between fill mode and line mode to draw the triangles.
  - R: Resets the window size to 700 X 500.
  - A: Resets the window size to 600 X 600.
  - F10: Saves a screen shot of the graphics window to a png file.
- If the control key is down:
  - A: Selects all rectangles.
  - Q: Deselects all rectangles.