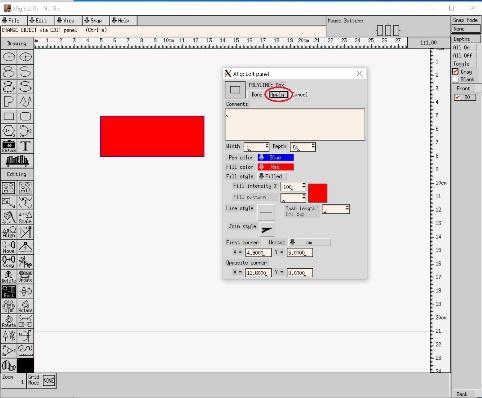
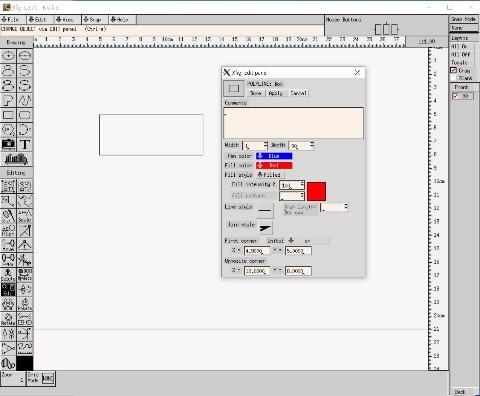
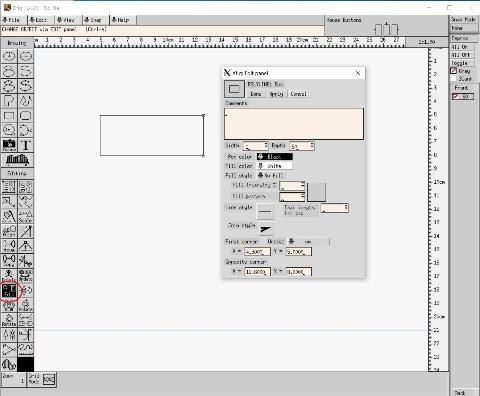
**Software Change Request**

| **Software** | **Baseline Version** | **Feature Name** | **Difficulty** |
| --- | --- | --- | --- |
| Xfig | 3.2.8a | Quick Coloring | Medium (Est. 7 files; 312 LOC) |

***Current Behaviors:***

Currently Xfig allows a user to specify both the fill color and the border color at the time of the object’s creation. If a user wishes to change the object’s fill and border color after creation, they first have to click the edit shortcut button and then on the object to pop up an edit menu. Figure 1 illustrates the current behavior through an example figure (a rectangle).



*Figure 1: The object property edit window (left), set the fill color in red, and border color in blue (middle), apply the color setting (right)*

***Expected Behavior:***

The current feature needs to be enhanced, since it is ineffective, especially when the user needs to specify the same fill and border colors for multiple objects. To improve this process, add a paint tool button in the Editing Mode Panel on the left hand side of Xfig. After selecting a fill color and a border color in the indicator panel for this feature, right-clicking an object will fill it with the selected fill color, and left-clicking will set it the selected border color.

***Solution Hints for Instructor:***

This task requires modifying and understanding both the indicator panel and button panel. Adding a button is done by adding an entry to the **mode\_switches** array in w\_modepanel.c, but adding an icon to the button is done by adding code to w\_icons.c and w\_icons.h. Controlling what buttons appear in the indicator panel is done by changing the indicator mask option for the entry in **mode\_switches**. For further information, the **mode\_switch\_struct** struct (the type of **mode\_switches**) is defined in w\_modepanel.h. The indicator panel works by setting various global variables which are used in the coloring code.

There are also some minor constants that can be added, which consist of the mode value (in mode.h), and the indicator mask (in w\_indpanel.h). This is not required for the feature to function properly, but preferred for consistency since it is done for all other drawing and editing modes.

The code for implementing the color changing feature is fairly self-explanatory. The corresponding functions are assigned to mouse buttons. When an object is clicked, a copy is created with the changed color, then is replaced with the original. This is all done with built-in functions that handle lower-level functions. Another feature added in this solution is the ability to set the fill pattern as well as the color. This is done in a similar fashion to changing the color.