CMSI 370-01

Interaction Design

Fall 2015

Assignment 1211 Feedback—Direct Manipulation Widget

Dustin Kane *ranneyd / dustinpkane@gmail.com*

Notes while running (asterisks indicate major observations):

- Nice, clear demo! +(*2b*, *4a*)
- Bounds checking is good. (+*4a*)
- Rubberbanding has a loophole—it's not possible to cross boundaries. Typical rubberbanding allows that. (see the mouse-driven box sample code) (*2b*, *3a*, *3b*, *4a*)
- Integration with front end is well-done! Though the lack of cross-border rubberbanding makes the chat window seem to disappear **:)** (*2b*)

Code review (asterisks indicate major observations):

1. Tabs in both CSS and JavaScript files (in the *front-end* sources; actual plugin code is clear). (*4c*)
2. Ideally this is in a CSS file, but it's a demo, so fine. (*4b*)
3. Nice; liking the callback. (+*4b*)
4. So this is where you would update the logic to support cross-border rubberbanding. I can understand that this did not occur to you because your foremost use case is to resize a chat window, and windows don't have this cross-border behavior. But drawing programs do, for many generic shapes. Because your plugin is meant for general use and not necessarily for just window resizing, it would be good to have this as an option, with some kind of sensible default in case the dev doesn't care. (*2b*, *4a*)
5. I would move these comments below the `else`, just to preserve the integrity of the overall statement. (*4c*)
6. An `else` clause is still in the same statement as the preceding `if`, so don't break them up. (*4c*)
7. In terms of overall code presentation, the code is mostly readable. There are choices that I would have made differently (and you've seen that feedback before) but at least you are largely consistent and I can respect that. (*4c*)
8. Nicely done, that's how the plugin should be invoked. (+*4b*)

2b — + …Rubberbanding would be the ultimate in generality, but overall not worth a hit.

3a — +

3b — +

4a — +

4b — +

4c — +

4d — +

4e — +

4f — +