CMSI 370-01
INTERACTION DESIGN
Fall 2014

Assignment 1204b Feedback

This feedback is effectively a code review; to keep it concise, notes are provided as a numbered list. Inline comments in your code illustrate an example of the numbered item (there may be more). Asterisks (*) denote high-priority issues. If any code review item is unclear or if the resulting proficiencies are not self-explanatory, please ask me.

Joaquin Loustau *joaquinloustau / joaquinloustau@gmail.com*

1. Both of your demos show your concept nicely, first in the abstract with *widget-demonstration.html* and then more concretely with *widget.html*. Because the latter is so clearly the way you plan to integrate this into your RPG, I am a little surprised that you didn't just do that. As it stands that aspect remains undone. (*2b*, *3a*, *3b*, *4a*)
2. Meanwhile, those filenames provide no notion at all of what your widget does! (*4c*)
3. And while we're on the subject of files, I must admit, it is very hard to glean your process based on the files. On your repository you have *widget-demonstration.html* and *widget.html*. For JavaScript there is *widget.js* and *plugin.js*. Then on *my.cs.lmu.edu* you actually have *files that are not on your repository*. There's *plugin-demonstration.js*. The containing directory is *touch-screens*—again absent from your repository. And now also *plugin-old.js*. These are all signs of a disjointed workflow that can only be counterproductive. Make sure your file structure is self-descriptive; don't work outside of version control; and ideally work on one machine, with the other one serving purely as a final publication platform (i.e., don't edit on that machine). (*4c*, *4e*)
4. Your element structure is completely hardcoded (this includes identifying elements by `id`—remember that there can only be one of those on a web page)—this limits you from all kinds of things. There is clearly the concept of a destination for your plugin (a "destination box" in your current code, albeit as an `id`) as well as a source…stick with those general ideas Your plugin manages the transport of items from one place to another; it should let the calling application determine the details. (*2b*, *3a*, *3b*, *4a*, *4b*, *4c*)
5. Your repurposing of the boxes code is fairly clever; just make sure that when you repurpose, you repurpose *completely*: names, elements, logic, etc. (*4c*)
6. Yikes, and how about that…*widget.js* has nothing to do with anything, it turns out. Good thing too, because that file is peppered with tabs **O_o** (*4c*)
7. OK, so the action is all in *plugin.js*; in there the main issues are assorted hardcodes and some incomplete repurposing (already mentioned). In addition, you don't need `BoxesTouch` to be visible anymore: the plugin wrapper takes care of that now. There are some RPG-specific names/references in there too—those should stay with the RPG, somehow passing into the plugin as an option. (*4b*)

2b — |
3a — |
3b — |
4a — |
4b — /
4c — /
4d — |
4e — / …This is affected by the haphazard-seeming file organization and naming.
4f — / …Nothing really as of the due date, with widget work starting 6 days later.