CMSI 370-01
Interaction Design
Fall 2015

Assignment 1211 Feedback—Direct Manipulation Application

Flanders Lorton *florton / freeflowingvortex@gmail.com*

Notes while running (asterisks indicate major observations):

- Gravity seems to be in the opposite direction? (*3b*)
- Bounds checking goes slightly beyond the drawing area. (*3a*, *3b*, *4a*)
- Flick velocity seems inconsistent; sometimes it is really fast, sometimes not. (*3b*, *4a*)
- Touch feedback is lost. It was already there, and it helped tell the user when their finger was on a box—why was it removed? (*2b*)
- Something about the device motion doesn't feel completely natural…at low accelerations, the boxes just keep on moving slowly. Maybe it's solely because no friction is being applied (and if so, that isn't a problem, since it wasn't really required). We'll see. (*3b*, *4a*)

Code review (asterisks indicate major observations):

1. Yay, no tabs **:)** (*+4c*)
2. Ah, no wonder the feedback is gone—the CSS reference was removed. This is a questionable decision—it intentionally made the user interface worse. Remember this is still an interaction design course. (*2b*)
3. For function definitions, place a space between `function` and the argument parenthetical. Think of it as a function statement, but without the name in between…there's still a space there, right? (*4c*)
4. Here's where the bounds checking is slightly off—for the right boundary, *use the right side* of the box. It's simply the left side plus the box's width. And as is, the code is using `20` as a magic number—it should be given a meaningful name. (*3a*, *3b*, *4a*, *4b*)
5. And it turns out the bounds check on the left is also easy to fix. Don't check against a hardcoded zero—check against the drawing area's left side! In a sense, this is more serious, because the drawing area might not necessary be aligned with the left border (it already isn't). (*3a*, *3b*, *4a*)
6. Same pattern applies to top and bottom boundaries; see notes 4 and 5. (*3a*, *3b*, *4a*, *4b*)
7. Here's why the flick feels uneven: the code is setting the velocity *independent of when the last move took place*. This means that the distance being computed here might have varying time durations—and thus the velocity is inaccurate. (*3b*, *4a*)
8. You've got two `if` clauses that lead to the same result—combine them. It's essentially a disjunction of the two conditions. That may make things long, but it's better than duplicated code, and if the condition length is really bothersome, then you can turn it into a function. (*4b*)
9. *** Here is another negative consequence of removing the CSS—this code! This violates MVC separation. If you want to customize how an element looks, *do it in CSS*. That makes things easier to maintain. (*4b*)
10. *** Aha, here is the source of the unnatural-feeling device motion: the code is not treating the acceleration like acceleration. Remember that acceleration is a rate of change on the velocity. Thus, it should not be used as an additional term in changing the position—*it should be used to change the velocity*. (*3b*, *4a*)
11. These conditions very closely resemble the conditions when moving the box. You should find a way to refactor these into a function that both pieces of code can call, so as to keep the bounds logic in a single place. (*4b*)
12. Third time's the charm! Yes, this bounds logic should definitely be unified. (*4b*)

Assignment 1211 Feedback—Direct Manipulation Application

13. More magic numbers—what do these values mean? Call them that. (*4b*)

2b — | …Removing the feedback is nominally not a big deal, but seeing as this had the secondary consequence of causing MVC violation (note 9), this can be considered to be an incorrect design decision.

3a — | …Generally good, but that bounds check is both glaring yet easy to fix, with sufficient information on the drawing area and its relationship to the boxes.

3b — | …Also decent but ultimately imprecise, with respect to the flick velocity and the response to device motion. This is actually a classic example of why direct manipulation is hard—some kind of code allows it to work in some way, but unless the response is fully accurate, the behavior can feel "off." That is what's happening here, and the code notes indicate how they can be fixed.

4a — + …Overall functionality is decent though; the application works without critical problems. We will chalk up the "high precision/accuracy" issues solely to *3b*.

4b — / …MVC separation issue truly comes to roost here. All alone this might have been a |, but the outcome is also hurt by the magic numbers and repeated code found elsewhere (notes 4, 6, 8, 11, 12, 13).

4c — +

4d — | …Some additional reading on how the acceleration values should be used may have helped.

4e — +

4f — + …Minor copying done after the due date, but just a copy so no problem.