CMSI 370-01

Interaction Design

Fall 2015

Assignment 1029 (due 1103) Feedback

Note that, as a condition for the due date extension, you were still expected to commit something by 1029. This will factor into your *4f* proficiency.

Trixie Roque

trixr4kdz / euqor00@gmail.com

Notes while running (asterisks indicate major observations):

- The layout looks like it needs better spacing, but it is at least fairly manageable. (*3a*, *4d*)
- Hmmm, top posts can be pretty lengthy depending on the tumblr that's being viewed. (*3a*)
- Core functionality is there but I think data display can be much better. In particular, blog details can have better typography and layout. And top posts + search results can use some refinement (ditch the bullets!). (*3a*, *4d*)
- Nice integration between search results on the right and blog section on the left! Although shouldn't existing data clear up when a new blog is chosen? Also, I don't think radio buttons are the best control for choosing the new blog. (*3a*, *3b*, *4a*)

Code review:

1. *** You indented with spaces on HTML, but not JavaScript and a few snuck into CSS. (*4c*)
2. *** The `center` tag is obsolete—note how it pertains to a *view* rather than a *model*. All view properties should be done in CSS. The same goes for view-specific attributes like `align`. In Bootstrap, the `text-centered` class takes care of most centering cases, and you can always use custom CSS for all others. (*4b*, *4d*)
3. The vertical spacing for the button and error message aren't quite right—the solution here is to put each of them in their own `form-group` elements, siblings of the one containing the search field. That is the intent of that class—to represent a distinct component of an overall form. (*3a*, *4d*)
4. You can think of this as a "form consisting of a single button," and thus it should be marked up with the form classes. You will notice that this will result in better spacing. (*3a*, *4d*)
5. I appreciate the explicitly descriptive labels on these buttons, but their relative length calls for a layout that accommodates them better. The default left-to-right sequence doesn't work out because the line breaks don't space out nicely. Maybe a strict vertical column? Don't fill the entire width, but maybe 3/4ths of it. Just something that cleans things up. (*3a*, *4d*)
6. Here's one idea: because these are, for all intents and purposes, distinct and autonomous segments, maybe using Bootstrap's `panel` framework or some other type of spacing will keep things cleaner? Something to think about. Although parts of it might be considered the realm of a designer (visual, not interaction), in terms of just basic spacing, coloring, and bordering, that is within our reach. (*3a*, *4d*)
7. Dig into Bootstrap a little more and you'll find their "semantic" classes—i.e., CSS classes that set a color based on *meaning*, such as "error" or "danger." This will reduce your custom CSS a little. (*3a*, *4d*)
8. Similarly, the aforementioned `panel` classes may obviate the need for this set of custom CSS properties. Not that custom CSS is completely unnecessary; it is just that what you have here so far appears to be also taken care of by Bootstrap, if you look further into their documentation. (*3a*, *4d*)
9. *** This is a global variable! Does it *really* have to be global??? And such a generic name too—that is very dangerous. (*4b*)
10. 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*)

CMSI 370-01

INTERACTION DESIGN

Fall 2015

Assignment 1029 (due 1103) Feedback

Note that, as a condition for the due date extension, you were still expected to commit something by 1029. This will factor into your *4f* proficiency.

11. Space after `if` (and most other reserved words) please. (*4c*)
12. In JavaScript, triple equals (`===`) is the appropriate equality operator. Double-equals is not sufficiently strict and is ultimately a language design flaw. (*4a*, *4c*)
13. For this type of event handler, you can just do: `$("#tag-button").click(tagSearch);` (*4b*)
14. *** Another global! Nooooo… (*4b*)
15. But functionally, a fun idea **:)** +(*4a*)
16. Semicolons are indeed optional in JavaScript, but stay consistent. (*4c*)
17. Why the shift to a `function` statement here? For consistency of notation and overall accuracy of semantics, we *do* prefer `var function_name = function () {` … when defining functions. (*4c*)
18. This is a subtle point, but you can handle it: in principle, it is not appropriate to use a conditional expression for branched execution. Strictly speaking, this should be an `if/else` statement. Key distinction is in the terminology: one is an *expression* but the other is a *statement*. This is a source of lively debate; we can talk about it sometime if you like. (*4b*)
19. Unless adjacent to same-sided parentheses, have a space before and after braces. (*4c*)
20. Sorry, but `optradio` is not a sufficiently descriptive name for this set of radio buttons. (*4c*)
21. Notice here that you have multiple calls to the same jQuery object. When you see this pattern, you can use *currying* to make the code more compact, for example: (*4c*, *4d*)

```
$("#show-posts ul")
    .append('<li class="post">' + '</li>')
    .append(img);
```

22. When performing the same operation over a group of elements, you can concatenate their selectors with commas to turn this into a single line: (*4c*, *4d*)

```
$("#blog-name, #likes-results, #posts-results").text("");
```

23. In nearly all situations, an opening left brace should not be on a line by itself. (*4c*)
24. Note how you are calling a function with the exact same arguments as the callback—this means you can directly set the callback to *be* that function! (yes, this is a corollary of note #13) (*4b*, *4c*)
25. Make use of JavaScript truthiness/falsiness—avoid this comparison, and if this is somehow needed specifically, explain it in a comment. (*4a*)
26. An `else` clause is still in the same statement as the preceding `if`, so don't break them up. (*4c*)
27. Why not use `forEach` here? (*4b*, *4c*)
28. Keep your code lines within a finite maximum; 120 characters max per line is typical these days. You can linebreak HTML tags between attributes. (*4c*)
29. You are returning a variable that you *just* defined…why not skip the variable and simply return the expression that you assigned to that variable? (*4b*, *4c*)
30. Why is this function outside of the closure? That makes it global. (*4b*)

CMSI 370-01

INTERACTION DESIGN

Fall 2015

Assignment 1029 (due 1103) Feedback

Note that, as a condition for the due date extension, you were still expected to commit something by 1029. This will factor into your *4f* proficiency.

3a — **|** …Overall decent and clean, just needs some clarity refinements, most of which can be accomplished by leveraging Bootstrap more deeply.

3b — **+** …Lots of good event-handling here.

4a — **+** …Great functionality!

4b — **|** …Overall decent, but missing some finer points.

4c — **|** …Also by and large readable, but there are enough hiccups (naming, spacing, etc.) to take this down.

4d — **+** …Some fun exploration of the Tumblr API, plus the successful use of a web service relay is not going unnoticed. You could have dug into Bootstrap more, but the aforementioned successes with outside information more than make up for that.

4e — You have some good pacing there, and messages are short but mostly descriptive. That works **:)** (**+**)

4f — Started before 1029, submitted on time (consideration extended due to SACNAS). (**+**)