Design Exchange UI

Sean Zhu

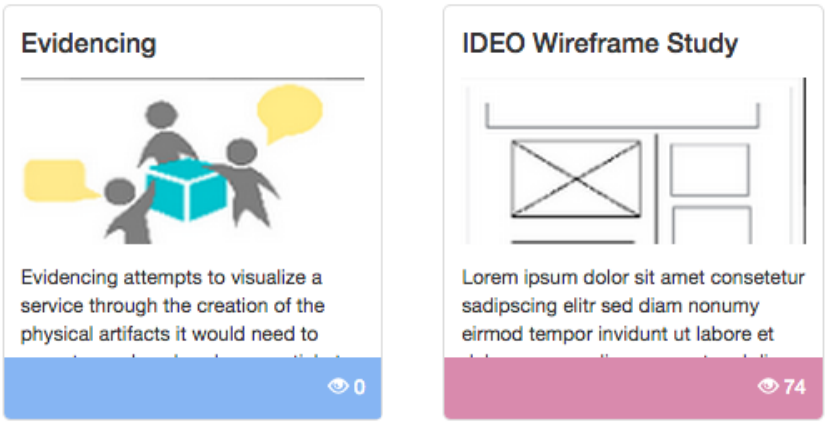
May 8, 2015

- Major Field: Design
- SID 23893798
- Mentors: Prof. Alice Agogino, Graduate student Cesar Torres
- This abstract may be posted on the ME website.

The Design Exchange (DX) is a website that allows various designers to exchange ideas, including design methods and case studies. There are a wide variety of these ideas, and so this information needs to be categorized in an intuitive and systematic way. Our work this semester focuses on improving the DX experience for both users and developers of the website.

Users. From a user perspective, we improved some of the features that DX users worked with most.

- Cards. DX is essentially a collection of design methods and case studies ("entries"). In order to find the entry that a user wants to see, the user must typically browse through a listing of entries, displayed as a card. Whereas previously the card only contained text and no graphics, we improved the card style to show more visual cues: a thumbnail of the entry is shown, and the bottoms of cards are color-coded to indicate whether they are design methods or case studies.



- Search. There are three ways to access entries: through a direct link, by browsing categories, and by search. Search is typically the fastest way to access an entry that a user has some

information about but does not have a direct link. We expect search to be the most common way for entries to be accessed, and so the search UI must be intuitive and fast. This semester we re-designed the search UI to show filters in a hierarchical sidebar. In the future, we plan to allow search results to be refreshed without reloading the entire page.

- Text descriptions. Whereas previously the description bodies of entries allowed only plain text, we now allow descriptions to contain Markdown, to better convey their meaning.

Developers. We also want to make the site easier to maintain and develop. DX is in a startup phase, and so features are subject to quick and drastic changes. For example, DX originally hosted discussions as well as design methods and case studies, but after it failed to gain a community, the discussion feature was removed. As another example, code for presenting design methods and code for presenting case studies has been maintained in parallel but separately, but it has become increasingly apparent that these two kinds of entries would benefit from being handled by the same code, as they largely have the same appearance.

However, unlike most startups, DX is unique in that many hands have touched the codebase, and not all developers received training on all the coding conventions used (Rails, MVC), nor can we reasonably expect or require all future developers to receive this kind of training. Therefore, we must make it easy for a few experienced developers to convert code to match the conventions used, and code that is written according to conventions must be easy to read and maintain for newcomers. As a result, we made some changes to the infrastructure of the code:

- HAML and CoffeeScript. HAML is an indented language that translates directly to ERB. CoffeeScript is a language that compiles to JavaScript, which contains conveniences for commonly used JavaScript idioms. We believe that adoption of these languages will make it written code more consistent, as there will be only one typical way to write each thing.

- CSS, not Sass. In the spirit of the above, we considered whether to convert CSS to Sass, a nested language that compiles to CSS. While typically seen as analogous to HAML and CoffeeScript, Sass is not line-by-line translatable to CSS, due it its nested nature. This difference may not only confuse users who are accustomed to CSS, but also make rules harder to write, as exceptions to nested rules may not always be able to be written in the nested context.

- Source maps. When using a web browser to debug CoffeesScript code, users will typically see traceback pointing to lines of compile JavaScript. Source maps, a relatively recent feature in major web browsers, allows lines of JavaScript to be able to be mapped to lines of source CoffeeScript, so the developer never has to see JavaScript code.

Conclusion. We hope that these changes to DX will make the site easier to use for everyone. In the future, we will continue to monitor users' and developers' feedback to these features in order to further improve the DX user experience.