COMS W4170: User Interface Design—Fall 2013

Prof. Steven Feiner
Date out: September 19, 2013
Date due: October 3, 2013

Assignment 1: The Beginning

Your first assignment has three parts, intended to help you get familiar with CogTool v1.2.2, HTML5 (actually the HTML4 subset), JavaScript, CSS and your chosen IDE, and some of the concepts that we have covered in class:

- Perform a heuristic usability analysis of LionSHARE, the Columbia University job portal for students and alumni.
- Use CogTool to perform a KLM analysis of two ways to accomplish the same goal in Google Chrome.
- Use your chosen IDE to implement a simple application in HTML5, JavaScript, and CSS.

Part 1: Heuristic Analysis (40%)

The first part of your assignment is to perform a usability analysis of LionSHARE, the Columbia University job portal for students and alumni, using Nielsen's usability heuristics (J. Nielsen, Ten Usability Heuristics). Everything you find after logging in to LionSHARE is fair game to include in your analysis (for better, and for worse).

In doing this part of the assignment, please follow the approach described at http://www.useit.com/papers/heuristic/heuristic_evaluation.html, but with a *single* evaluator—you! You should begin by trying to identify at least four specific problems caused by LionSHARE not satisfying the usability heuristics. For each of your four (or more) problems, (a) describe it, (b) explain how it does not meet one or more of the heuristics, and (c) rate the problem's severity, using the 0–4 scale explained at http://www.useit.com/papers/heuristic/severityrating.html.

Next, for each of Nielsen's ten usability heuristics, discuss how well LionSHARE meets that criterion. Here, you should refer back to the specific problems you identified earlier, when appropriate, but you can also mention any other problems (and positive features) you might find. Please write one or two paragraphs for each of the ten usability heuristics. In discussing each heuristic, you are also welcome (but *not* required) to address some of the issues discussed in Shneiderman and Plaisant, Chapter 4; for example, you could consider some of the questions in the questionnaires it discusses. (If you decide to do this, you can get a free look at an older version of the QUIS in http://hcil.cs.umd.edu/trs/87-11/87-11.pdf.).

Please feel free to include screenshots along with your text. Your discussions of at least four usability problems with LionSHARE *and* your analyses of how well LionSHARE satisfies each heuristic, should together take about three pages of text (not counting screenshots).

Do not forget to consider LionSHARE in the context of other applications you have used and the normative UI standards set by the OS on which you run the application (heuristic #4). That is to say, do not simply consider LionSHARE as an isolated artifact.

Part 2: Keystroke-Level Model Analysis (20%)

In the second part of your assignment, you will use CogTool to mock up and explore the estimated performance of two different ways (differing in the actions that the user performs) of bookmarking a web page in Google Chrome. You should assume that each way starts from the same initial conditions in which only the browser view is open and the current page has not yet been bookmarked. You don't need to specify the stored folder (just use the default folder).

To get up to speed with CogTool, you should get acquainted with its documentation, including reading through the CogTool User Guide, and trying out the Step-by-Step Tutorial. For this part of the assignment, you should create a CogTool Project containing a single storyboard Design that uses background screen images that you have grabbed from Google Chrome itself (using the screen capture facilities in your operating system), and the necessary Standard Widgets and Transitions created with CogTool, as we demoed in class. Each of the two ways of bookmarking a web page should be encoded as a separate Task in that Design, with a Script that accomplishes it. Please lay out the Frames in your Design neatly to make it easy for you (and us) to understand how they are connected.

After using CogTool to compute the predicted step-by-step timing for each of the two Tasks, please write a short (one paragraph) comparative explanation of the computed predictions, based on the class notes and the Google Chrome documentation. Your explanation should describe in English why the predicted times were either roughly similar or different.

Part 3: Twitter Search (40%)

The purpose of the third part of the assignment is to get you up to speed with HTML5, JavaScript, JSONP, and your IDE of choice by creating a simple web-based UI for specifying and refining a search on Twitter. To accomplish this, you will be using the Twitter API v1.1, specifically:

- Twitter Search (and see this documentation)
- Twitter Timeline
- Twitter Tweet
- Twitter Geo Search

Your user should be able to specify a query using arbitrary keywords *and* at least the following set of operators (used in the following list with example arguments):

- place:247f43d441defc03 (about the place with Twitter ID 247f43d441defc03)
- since:2013-09-15
- until:2013-09-17

Support for additional search operators is welcome. Your application should display a set of 50 tweets at a time, and provide the user with the ability to retrieve all tweets satisfying the query, as they interact with your application. Your application should display information that you think would be interesting to someone searching for tweets, using a format and layout of your own choosing. For each tweet you display, there should be some way for the user to "drill down" to request additional details about (a) the user who made the tweet and (b) the tweet itself. The details that can be requested and displayed are up to you.

Your application should also remember and provide access to the last ten search queries (not

result sets) that your user has made in the current session. (Note: You are *not* being asked to make this information persist across sessions.)

The description for part 3 is intentionally high level. Your design should take into account the usability issues and heuristics discussed in class and in your readings thus far, and will be evaluated in that context, while factoring in the limitations that we have built into the specification to make this assignment more manageable. A full-featured Twitter search application would have much more functionality than the version described here, but we're asking you to keep it simple!

Please note that your grade for part 3 will be based in part on your analysis of the design decisions that you made (included in your submission, as described below).

Submission

Please put the heuristic analysis of part 1 and your one-paragraph comparative analysis of part 2 in a single document, with each part clearly labeled. The document should include any screenshots for part 1. (You're also welcome to include screenshots for part 2 if you think they would be helpful.) When referencing a screenshot, please specify clearly in your discussion the image to which you are referring (numbering your figures is a good way to do that). If your CogTool Project file has more than a single Design with two Tasks, please clarify by name, in your paragraph, which Design and Tasks we should be looking at. The preferred method of submission is as a .pdf file; however, a Word .doc[x] file (as a last resort) will also be accepted.

For part 2 of the assignment, also submit the CogTool project file (*.cgt).

For part 3, first make sure that you include a README file that contains (a) your name, UNI, and email address, (b) an explanation of any special instructions for running your application, and (c) a paragraph justifying your user interface design decisions, and, optionally (d) a description of any additional functionality you have implemented beyond that required for the assignment.

Then create a .zip file containing all the files needed for your application. Next, just to make sure you've done this right, move your .zip file to a temp directory, unzip the file, and check whether everything is there. (Your TAs are not going to be sympathetic to explanations that something accidentally got left out of your .zip file.)

Now you're ready to submit, using CourseWorks, in response to the assignment "Assignment#1".

Note: If you wish to resubmit before the deadline, you can do that. We have set CourseWorks to allow unlimited resubmissions, and we will grade your latest on-time submission.

Hints

You will need to register to use the Twitter API. Typical of commercial APIs, Twitter implements rate limiting, and you should be aware of the limits. It's certainly possible for a bad design (or buggy implementation) to max out your quotas, and spend a lot of your user's time doing that.

Please do not implement or use any kind of server-side technology that your application will require to run. This means that no PHP, Ruby, .NET, or similar technology should be added to your code, and we should not be required to set up anything to run your code, either. In other

words, you and we should be able to run your application using only the files you provide, on a modern browser (e.g., Mozilla Firefox or Google Chrome). And, please be sure to test your application outside of the IDE in which you develop it.