COMS W4170: User Interface Design—Fall 2018

Prof. Steven Feiner
Date out: September 20, 2018
Date due: October 4, 2018

Assignment 1: The Beginning

Your first assignment has two parts, intended to help you get familiar with the basic tools we'll be using for web-based UI development, as well as some of the concepts we have covered in class. In this assignment, you will:

- Perform a heuristic usability analysis of an existing website.
- Use your chosen IDE or editor to implement a simple application in HTML5, CSS, and JavaScript that interfaces with an existing API.

Please note that this is an individual project. You should not be working together with anyone else on any part.

Part 1: Usability Analysis (50%)

In class, we discussed Nielsen's 10 Usability Heuristics. For Part 1 of this assignment, you will be conducting a usability analysis of the Columbia Vergil website.

Everything you find, starting with the first page, is fair game to include in your analysis (for better or worse). Please follow the approach described at How to Conduct a Heuristic Evaluation but with a *single* evaluator—you!

Your tasks are:

1. Find **five specific usability problems** with the Vergil website. For *each* problem:
    a. Describe the interface behavior you observed.
    b. Explain why this behavior is a problem for the user. You should refer to at least one of Nielsen's heuristics.
    c. Include at least one screenshot illustrating the problem.
    d. Rate the problem's severity, using the 0–4 scale explained at Severity Ratings for Usability Problems.

Note: You should write a full paragraph for each problem.

2. Conduct a **full heuristic analysis** of the Vergil website. For *each* of the ten heuristics,
    a. Briefly summarize the heuristic in your own words.

b. Explain how well you think the Vergil website meets this heuristic, and why. When appropriate, you should refer back to the specific usability problems you identified earlier. You can also mention any other problems or positive features you find. You are also welcome (but not required) to address some of the issues discussed in Shneiderman et al., 6th ed., Chapter 5.
c. To illustrate how well the heuristic is satisfied, you can optionally include screenshots. (This is *highly* recommended if it will make your explanation easier to understand.).

Note: You should write a full paragraph (or two) for each heuristic, written in a way that sounds like you have read and thought about the material linked to this section.

Remember to consider the Vergil website in the context of other applications you have used.

As described below, your submission for this assignment should include a single **PDF document** for Part 1 called **Part_1.pdf**.

Part 2: Marvel Character Search (50%)

In Part 2, you will use HTML5, CSS, JavaScript, and your preferred IDE or editor to create a simple browser-based application that uses the Marvel Comics API to search through the Marvel comic database.

Please begin by reading Getting Started and registering for a free API key. That will allow you to experiment with the the Marvel Interactive API Tester as you read through the interactive documentation integrated with it.

Your application should allow the user to query for *comics* and display 10 results at a time. Each result should include the first Marvel *character* (in alphabetical order) that appeared in that comic. You should present the *name*, the *description*, and a *thumbnail* image of the character. (Note that some characters do not have a thumbnail, in which case you should handle this case gracefully.) Your user should be able to specify the following filters when querying comics, *each of which should have a reasonable default*:

- Enter a *title* OR initial letters of a title
- Enter a *startYear*
- Select the issue *format* (e.g., comic, digital comic, or hardcover)

When formulating a query, the user should also be able to choose how to order the return results, either by ascending/descending *title, issueNumber,* or the default order.

If more than 10 comics satisfy the query, the user should be able to move forward (and backward) sequentially in the results; for example, to view the next or previous page of up to 10 comics. However, you do not need to make it possible for the user to skip forward and backward over results (although you may optionally do this if you want).

Please note that one call will return at most 100 results—to obtain more, you will have to make multiple calls, using the *offset* (and *limit*) parameters, as needed. Further, note that the Marvel Comics API will limit you to at most 3,000 calls a day. So, you may wish to consider caching some portion of the results to avoid making unnecessary calls when a user moves forward or backward through the results of a query.

Please feel free to explore the entirety of the API. However, you should concentrate on the comics and comicsCharacters endpoints.

You are welcome (but not required) to use any of the following frameworks:

- jQuery
- HTML5 Boilerplate
- Bootstrap
- Foundation

This assignment description is intentionally high level. Your design should take into account the usability issues and heuristics discussed in class and in your readings. It will be evaluated in that context, while factoring in the limitations we have specified to make this assignment manageable. We're asking you to keep it simple, but well designed!

Your submission should include a README.txt file that contains instructions on how to run your app, including any dependencies that need to be downloaded. It should also describe your design process and the decisions underlying it, including the heuristics you considered. Your grade for this part of the assignment will be based in part on your explanation of the design decisions that you made. Please note that our determination of how well your design worked will be influenced by that explanation.

If your submission is only partially functional, or works only on certain web browsers, please say so in your README.txt to receive partial credit.

Submission Instructions

You will submit this assignment on CourseWorks in response to Assignment 1.

Your submission should be a compressed (zipped) folder called **<your-uni>_HW1.zip**. For example, Steve's uploaded submission would be skf1_HW1.zip.

The submission folder should include:

- A PDF for Part 1 called **Part_1.pdf**
- A directory (folder) for Part 2 called **Part_2**, containing:
    - **index.html**
    - **styles.css**
    - **script.js**
    - **README.txt** (see description above)
    - Any other files, subdirectories, or libraries needed to run your app

Just to make sure you've done this right, please inspect your submission folder carefully to check whether everything is there. And, after you upload your file to CourseWorks, please download it to a different directory on your computer and extract the files in it to make sure that they are the specific files you wanted to upload. Your IAs are not going to be sympathetic to explanations that something accidentally got left out or that you accidentally zipped up the wrong files or uploaded the wrong zip file. They will grade only what you uploaded, not what you intended to upload!!!

Hints

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 augmentation should be added to your code, and we should not be required to set up any kind of tool to run it. This does not mean that your application should not communicate with a server (this is how you make API calls, by sending requests); it means only that *we* should not need to set up a server to run *your* code. To reiterate, *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).

Please bear in mind that the runtime environment of your IDE may end up using a local server for debugging purposes. So, just to make sure, please also test your application outside of your IDE, without running such a server.

Please be sure to read (and practice) the last paragraph of the Submission Instructions. And don't forget that *you can use at most one late day for this assignment*.

Grading

Part 1	**Usability Analysis**	**50 points**
	Five usability problems	20 points
	Heuristic analysis	30 points
Part 2	**Marvel Character Search**	**50 points**
	Design	25 points
	Functionality	15 points
	Documentation	10 points
Total		**100 points**