CSE121B
Michelle Markham
Week 07
June 1, 2022

Assessing Online Sources

- What, do you think, made the difference between a good source and one that wasn't so good?
    - I believe that a good source provides supporting information, even a little background and reasons why something is ‘good’ or ‘bad’ or sometimes, just outdated. They will also clarify what is their ‘opinion’, vs. industry standard usage. They will use commonly understood language and indicate when they are using ‘synonyms’, or variations of terms and explain in plain terms any use case differences. This is helpful as things are constantly in ‘flux’.
    - I also feel that a good source stays current. They may have information that was great a few years ago, but they will update their content with current industry standards and improvements. Of course, there will be constant iterations, but for example, if I am watching a tutorial from a year ago, if they are still using the old ‘for loops’ and none of the newer array methods and arrow functions, I may learn a few things, but I am more apt to find another source.
    - Well structured code. You can tell if they have useful experience by how easily and they organize their code. It seems completely effortless. I’m mesmerized. I watch them code an entire site and I understand perfectly everything they are doing. I know it just takes practice, practice, practice, like an instrument. Your Uncle Bob may be able to teach a few riffs, and maybe you are naturally gifted… but if you are average and want to get really good, Bob may also pass on some bad habits that you will have to ‘unlearn’… so, there is a lot to learn from a lot of people, but over time you will find it best to learn from folks who were born to ‘teach’ and help you come away with a clear understanding of concepts and teach good, clean code, in a very natural way.

    - PS. I feel like I'm still wearing training wheels, but I'm learning.

- Have you learned to tell reputable sources from iffy ones?
- How have you begun to differentiate between reputable and iffy sources?
    - For me, it's more of a 'gut' feeling. Many factors, like those above, are good indicators. I've learned from newer YouTubers some helpful things. Some people just know how to teach clearly in a way that increases my understanding about the 'whole'.
    - StackOverflow CAN be useful, but you must often sometimes sift through a bunch of not so clear information. I will utilize it with discretion. There can be some interesting information shared by the experienced and there is also a lot of unclear code snippets that just increase confusion (and screaming brain cells).
    - MDN…hmmm, yes, it is the industry standard. One day I'll look up someone on YouTube who can give a tutorial with a cypher on how to translate it's pseudocode into functionality. I guess I just learn better from larger application examples. I really seem to struggle with interpreting what it is saying and coming away with a clear application that doesn't leave me with many follow-up questions, TBH.
    - W3 Schools – I really like these fellas. It is organized well and I can see things in action and mess around on the spot to test it in a way that spells out what it does and usually it makes the variations and usage pretty clear.
    - https://caniuse.com/
      This one is pretty obvious. I know I am way behind on all of the tools out in cyberspace and probably am doing a lot of things the hard way still, lol, but with this reference, you can check on industry standards and ask the important questions based upon your particular use case needs.
    - I personally find that just searching for a task that I want to accomplish or a specific term I wish to understand more deeply, gives me the best results. We all learn differently, but good teachers I've found on YouTube suit me best. I don't

limit myself by going directly to one person, because of the constant change. I do a search and may gravitate to teachers I have benefited from, but I choose current information over old, and in that sense, am not afraid to learn from someone new. I put my discretionary filter on with anything they say, do not take it for gospel, but will evaluate it based on corroborating information found in sources.

- If, for example, you were going to answer someone's coding question about this language, what should you do to be a reputable source?
    - Well, I would not consider myself a ‘reputable’ source, but I would help guide them based upon the criteria I’ve stated above. Do their research. Experiment. Find a tutorial from a good source to give you basics and background and ‘build’ knowledge from that place. Remember, nothing is set in stone, be open to change, because it’s going to happen. Learn how to ‘search’ intelligently and ask questions. Learn to ‘ask’ Google or whatever search engine you like first, because invariably, if you are asking, it’s probable been asked before. Get good at asking the right questions when you need to. (just note what frustrates you when you search for a question on StackOverflow… how could they ask better if you were going to try and help them with their issue? Ask that way).

- What online sources have you used and in what order would you rank them for quality and usability?
    - YouTube (discretion)
    - Google (discretion)
        - https://dynomapper.com/blog/410-top-25-websites-to-learn-to-code
            - Lists provided on random blogs like this gives me a good start. I scrutinize the choices, but there are many sources I recognize and can deem this a good place to start.
            - Ones here that I recognize and wouldn’t have issues recommending exploration to others:

- Khan Academy
- Code School
- edX
- Coursera
- FreeCodeCamp

  … you get the drift, wow, I'm going down a rabbit hole here. There are some REALLY great resources out there. I need to bookmark this page… or just trust I'll find it when I need it again.

OK. I stink at ranking. The thing is, there is no end to the internet and resources are abundant. You must use your 'inner compass' to go about it wisely. (I'm positive there is a Gospel analogy in here). If you don't know, ask. Do not get 'myopic' or 'dogmatic' in your efforts to learn effective coding. Be open to change because it is [const change = infinite;] … that's a conundrum there, lol.

Thanks, Bro Blazzard for a great introduction. This class may seem like it is a bit 'wanting'… in the sense that it can be easy to get through, but for me, who needed to get back into the 'swing' of things, it was not 'too much' stress. I know I could have done better. I've had more experience, but it's all jumbled up. I am still working on my 'focus'. I spent more than the allotted time, but I am still not 'practicing' as much as absorbing. I feel that I just need some 'wax on wax off' (Karate Kid reference) practice and trust in the process. Getting out of my head and solving problems on paper and putting it in code.