Author: William Z Chadwick
Date Created: 07-31-2022
Date Modified: 08-02-2022

Description: Contains PromineoTech's Front End Bootcamp Week 1 Quiz, research prompts, and my answers.

Quiz Prompts & Answers:

1. What is git? Why is it useful? What is the git workflow?

Git is a the most used version control system in the world. It is useful becausepeople who use it can save different versions of whatever they are working on (usually code) as they make changes, they can easily revert to previous changes or jump back ahead. They also can easily share work with each other and work on the same projects together. This is the underlying idea of the "git workflow."

Concretely how it works is very like this: You have an initialized git repository ("git init"). You add files which you will be working on to the repository by using the command "git add," followed by the names of those folders. Then you do some work and make some changes in those folders.

When you have made a few noticeable changes, you give the command, "git commit -m," including a short summary of your changes inside quotation marks after the "-m". After you press enter for this command, the next step, especially if you are done for th day or done making your changes, is to give the command, "git push -u origin main" (although if you are working on a different branch than main you will give a different last word to indicate the branch). This is probably the standard git workflow for most people working alone.

According to Colt Steele's Git and Github Bootcamp on Udemy (my source for this information aside from personal experience), there are actually several distinct possible workflows when collaborating with others. He demonstrates these in various videos so that the logic and flow feel concrete and become easier to understand. He says that none of these different work flows have offical names, so he calls them as such: 1. Centralized Workflow 2. All-Important Feature Branch Workflow. 3. The Fork and Clone Workflow.

Being a writer, and actually liking git a lot, I am tempted to write more about these different workflows - but I'll leave it off at that for now.

2. What data types do we have access to in JavaScript? What makes them each unique? What values can they hold?

From what I remember, the main data types in JavaScript are strings, numbers, boolean (true, false), and arrays. There may be a character data type, as in C++, and probably there are some more advanced data types for more advanced needs and uses (like the double data type in C++, and other data types for extremely large or extremely small numbers).

I am not completely sure how to say how each one of them is unique except to say

that each simply holds the type of data that it holds, and not another one. Strings can hold single or multiple characters; numbers can only hold numbers. Boolean only holds true and false (on/off, or 1/0 like binary). Arrays usually only hold one type of other data type, but collected together in order, starting at 0 as the first index in the array. I can't remember if JavaScript arrays can have multiple different data types included in them, but I feel like the answer is yes. I will have to look it up again.

Now that I have looked it up (https://www.w3schools.com/js/js_arrays.asp), I have found that arrays are actually objects, and that members of arrays can also themselves be objects. I didn't know this. So, I have been learning. (Previously, I have learned about JavaScript from Google's Grasshopper App [JS and practice thinking like a programmer], Mimo's Web Development Course [html, css, and JS], a Zenva course on JS, and some Odin Project lessons. A lot of this learning was familiarizing myself with concepts and syntax so that I had less anxiety and imposter syndrome about learning to code. The downside to this is that I have spent less time actively creating and problem-solving specific projects.)

3. What is your favorite thing you learned this week?

Honestly, I really enjoyed the things I learned in the git bootcamp on Udemy. At the end of the course, I learned about hexadecimal hash functions, also called "cryptographic hash functions" [in bash: "echo 'hi' | git hash-object --stdin"], which give a deterministic key of 40 hexadecimal characters. [Hexadecimals are digits which have a base count of 16, instead of 10 in our Arabic numerals, from 1-10, or 0-9.] I also thought that the lessons on "git rebase," and especially interactive rebase ["git rebase -i"], were interesting.