JavaScript Research Week 1 **9/30/21**
Lisa Smith

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

   Git is a software version control system. In fact, it is the most widely used modern version control system in the world. It was originally created in 2005, and is open source. Many, many software projects rely on git for their version control.

   Version control allows multiple developers to safely edit a software project through branching and merging. Branching is when a developer duplicates part of the source code (repository). That developer can safely edit and make changes to that code without affecting the rest of the project. Once the code is working, the developer can mereg that code back into the main source code to make it official.

   Git allows multiple developers to work on a project, using branches to isolate their changes, and then merging their changes in common repo (which has a main branch). These developers can create a private branch, and use that as a scratch pad to do their work, eventually pulling/pushing their branch code to the repository, where it can be evaluated, and merged safely. The repository is created in one of the services that help developers store and manage code. (e.g. BitBucket, GitHub, GitLab, etc).

   A git workflow is a recommendation for how to use git to do what you want to do with a software project. It will descript how to accomplish yoru work in a consistent and productive manner.

   A common git workflow might be as follows:

   - git clone repo_name
   - cd repo_name
   - git pull
   - git add .
   - git commit -m “comment concerning contents of commit”
   - git push -u origin main
   - git status

   The intended (best practice) workflow is:
   - Create a **private branch** off a public branch.
     $ **git branch newbranchname**
     $ git checkout newbranchname

   - Regularly commit your work to this private branch.
     $ git commit -a -m ‘Change to feature xyz [newbranchname]’

   - Once your code is perfect, **clean up its history.**
   - **Merge the cleaned-up branch back into the public branch.**

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

   JavaScript is a loosely typed language, and is dynamic. Variables in JavaScript are not assigned a particular type, and can be reassigned as needed.

Primitive values:

- **Boolean** type — a logical entity which can have two values, **true** and **false**.
- **Null** type — one value only — **null**
- **Undefined** type a variable that has not been assigned a value has the value undefined.
- **Number** type — a double-precision 64-bit binary format IEEE 754 value (floating point numbers). +Infinity, -Infinity and NaN (Not a Number). To check for the largest available value or the smallest available value, use Number.MAX_VALUE or Number.MIN_VALUE
- **BigInt** type —integers with arbitrary precision. A BitInt is created by appending n to the end of an integer or by calling the constructor. Use Number.MAX_SAFE_INTEGER. Operators allowed are +,*,-, **, and %
- **String** type — represents textual data. It is a set of “elements”, which is 0 based. The first element is at position 0.
- **Symbol** type — a unique and immutable primitive value, and may be used as the key of an Object property. Symbols are called “atoms”

Objects: a value in memory which is possibly referenced by an identifier. A collection of properties. Perperties are identified using key values. A key value is either a String value or a Symbol value. The two types of object properties are data property and accessor property (get and set).

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

Learning something new. Love figuring out a new language, and tying the new information to what I already know.

Sources:

https://sandofsky.com/workflow/git-workflow/
https://www.atlassian.com/git/tutorials/what-is-git
https://kinsta.com/knowledgebase/what-is-github/
https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Data_structures