JavaScript Research Week 1 **10/9/21**
Lisa Smith

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

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

This type of control system (version and source) 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 commit that branch to the master repository, and then merge 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 can be created in one of the web-based version-control and collaboration platforms for software developers. These platforms can also be viewed as a code hosting platform.

A few examples of web-based version-control and collaboration platforms are: BitBucket, GitHub, GitLab, etc. For our course work, we are using **GitHub**. **GitHub** is a graphical representation of our git repo, and this is true of these other sites as well.

A **git workflow** is a recommendation for how to use git to do what you want to do with a software project. It will allow a developer and prescribe how to accomplish your work in a consistent and productive manner.

Git Workflow: ***(PT, JavaScript Week 1 Videos***)

- **Pull**: Pull Down Changes (from collaborative repo) — git pull
- **Edit**: Make Changes (locally) — edit the program, test changes, etc.
- **Add**: Stage Changes (locally) — git add
- **Commit**: Take a snapshot of Changes (locally) — git commit
- **Push**: Push changes to Remote Branch (locally done, pushed to the collaborative repo) — git push
- **Merge**: Merge those changes in to the Master Branch — The Master Branch is production ready

One example of what you are going might look like this:

- 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: (***See sources below***)

* 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 dynamically typed language. Some might say it is loosely typed. Variables in JavaScript are not assigned a particular type, and can be reassigned as needed. JavaScript infers the type of data by the data assigned to that variable.

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. String are written with quotes, either single or double!
- **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).

Arrays: Written with square brackets, Array items are separated by commas, and are zero-based.

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:
PT Videos — FESD JavaScript Week 1
Git/GitHub:
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
JavaScript:
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Data_structures
https://www.w3schools.com/js/js_datatypes.asp