|  |  |
| --- | --- |
| **CREATING A REPO** | |
| **git init** | Running the git init command sets up all of the necessary files and directories that Git will use to keep track of everything. All of these files are stored in a directory called .git (notice the . at the beginning - that means it'll be a hidden directory on Mac/Linux). This .git directory is the "repo"! This is where git records all of the commits and keeps track of everything!  WARNING: Don't directly edit any files inside the .git directory. This is the heart of the repository. If you change file names and/or file content, git will probably lose track of the files that you're keeping in the repo, and you could lose a lot of work! It's okay to look at those files though, but don't edit or delete them.  config file - where all project specific configuration settings are stored.  Git looks for configuration values in the configuration file in the Git directory (.git/config) of whatever repository you’re currently using. These values are specific to that single repository.  Here's a brief synopsis on each of the items in the .git directory:  For example, let's say you set that the global configuration for Git uses your personal email address. If you want your work email to be used for a specific project rather than your personal email, that change would be added to this file.  description file - this file is only used by the GitWeb program, so we can ignore it hooks directory - this is where we could place client-side or server-side scripts that we can use to hook into Git's different lifecycle events info directory - contains the global excludes file objects directory - this directory will store all of the commits we make refs directory - this directory holds pointers to commits (basically the "branches" and "tags") |
| **git clone <path-to-repository-to-clone>** | The command is git clone and then you pass the path to the Git repository that you want to clone. Example: git clone https://github.com/udacity/course-git-blog-project |
| **git status** | The git status command will display the current status of the repository. |
| **REVIEW A REPO'S HISTORY** | |
| **git log** | The git log command is used to display all of the commits of a repository.  to scroll down, press j or ↓ to move down one line at a time d to move by half the page screen f to move by a whole page screen to scroll up, press k or ↑ to move \_up\_ one line at a time u to move by half the page screen b to move by a whole page screen press q to quit out of the log (returns to the regular command prompt) |
| **git log --oneline** | The git log command has a flag that can be used to alter how it displays the repository's information. That flag is --oneline |
| **git log --stat** | The git log command has a flag that can be used to display the files that have been changed in the commit, as well as the number of lines that have been added or deleted. The flag is --stat ("stat" is short for "statistics") |
| **git log -p** | The git log command has a flag that can be used to display the actual changes made to a file. The flag is --patch which can be shortened to just -p: |
| **git show SHA** | By supplying a SHA, is shows t that commit |
| **ADD COMMITS TO A REPO** | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |