Git Internals Workshop - Useful Commands

May 10, 2019

*NIX Commands

`mkdir <path>`	Creates a directory/folder
`ls <path>`	Lists the contents of a directory
`find <path> -type f`	Finds all files in a given directory
`cat <path>`	Prints contents of a file to `stdout`
`echo <message> > <path>`	Overwrites a file with the given message

`git` Commands

`git init`	Creates and populates `.git/` directory
`git hash-object -w <path>`	Saves an object to the Git object store and prints the hash
`git cat-file -p <object hash>`	Prints the contents of an object in the Git object store
`git cat-file -t <object hash>`	Prints the type of an object in the Git object store
`git update-index --add <path>`	Adds a file to the Git index
`git ls-files`	Lists the files in the current index
`git write-tree`	Saves the index as a tree in the Git object store and prints the hash
`echo <message> | git commit-tree <tree hash>`	Saves a commit pointing to a tree in the Git object store and prints the hash
`echo <message> | git commit-tree <tree hash> -p <commit hash>`	Saves a commit as above, but with (at least one) parent commit
`git log <commit hash>`	Prints all the commits previous to the one specified by the hash
`git log <commit hash> --all --decorate --oneline --graph`	Prints the log as a easily human parsable graph
`git update-ref refs/heads/<branch> <commit hash>`	Creates a named branched that points at the given commit
`git rev-parse <branch>`	Prints the commit hash to which the branch points
`git branch`	Lists all the branches in the repository
`git symbolic-ref HEAD refs/heads/<branch>`	Sets `HEAD` to point at the specified branch
`git update-ref refs/tags/<tag> <object hash or reference>`	Tags a git object with a lightweight tag
`git tag -a <tag> <object hash or reference> -m <message>`	Tags a git object with an annotated tag