**GIT INIT COMMAND**

The git init command creates a new Git repository. It can be used to convert an existing, unversioned project to a Git repository or initialize a new, empty repository. Most other Git commands are not available outside of an initialized repository, so this is usually the first command you'll run in a new project.

Executing git init creates a .git subdirectory in the current working directory, which contains all of the necessary Git metadata for the new repository. This metadata includes subdirectories for objects, refs, and template files. A HEAD file is also created which points to the currently checked out commit.

Aside from the .git directory, in the root directory of the project, an existing project remains unaltered.

By default, git init will initialize the Git configuration to the .git subdirectory path. The subdirectory path can be modified and customized if you would like it to live elsewhere. You can set the $GIT\_DIR environment variable to a custom path and git init will initialize the Git configuration files there. Additionally, you can pass the --separate-git-dir argument for the same result. A common use case for a separate. git subdirectory is to keep your system configuration "dotfiles" (.bashrc, .vimrc, etc.) in the home directory while keeping the .git folder elsewhere.

**FORK COMMAND**

A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. Creating a “fork” is producing a personal copy of someone else’s project. Forks act as a sort of bridge between the original repository and your personal copy. Most commonly, forks are used to either propose changes to someone else's project or to use someone else's project as a starting point for your own idea. Forking is at the core of social coding at GitHub.

A great example of using forks to propose changes is for bug fixes. Rather than logging an issue for a bug you've found, you can:

* Fork the repository.
* Make the fix.
* Submit a pull request to the project owner.

If the project owner likes your work, they might pull your fix into the original repository!