* Git is the industry-standard version control system for web developers
* Use Git commands to help keep track of changes made to a project:
  + git init creates a new Git repository
  + git status inspects the contents of the working directory and staging area
  + git add adds files from the working directory to the staging area
  + git diff shows the difference between the working directory and the staging area
  + git commit permanently stores file changes from the staging area in the repository
  + git log shows a list of all previous commits
* git checkout HEAD filename: Discards changes in the working directory.
* git reset HEAD filename: Unstages file changes in the staging area.
* git reset SHA: Can be used to reset to a previous commit in your commit history.
* git branch: Lists all a Git project's branches.
* git branch branch\_name: Creates a new branch.
* git checkout branch\_name: Used to switch from one branch to another.
* git merge branch\_name: Used to join file changes from one branch to another.
* git branch -d branch\_name: Deletes the branch specified.
* git clone: Creates a local copy of a remote.
* git remote -v: Lists a Git project's remotes.
* git fetch: Fetches work from the remote into the local copy.
* git merge origin/master: Merges origin/master into your local branch.
* git push origin <branch\_name>: Pushes a local branch to the origin remote.

The workflow for Git collaborations typically follows this order:

1. Fetch and merge changes from the remote
2. Create a branch to work on a new project feature
3. Develop the feature on your branch and commit your work
4. Fetch and merge from the remote again (in case new commits were made while you were working)
5. *Push* your branch up to the remote for review

Steps 1 and 4 are a safeguard against *merge conflicts*, which occur when two branches contain file changes that cannot be merged with the git merge command. Step 5 involves git push

A *remote* is a Git repository that lives *outside* your Git project folder. Remotes can live on the web, on a shared network or even in a separate folder on your local computer.

* The *Git Collaborative Workflow* are steps that enable smooth project development when multiple collaborators are working on the same Git project.