CHEATSHEET: GIT

- Git's user interface has become simpler in recent releases, so I strongly suggest installing version 1.8 or later. You can check which version of git you're running using the `git --version` command.
- Use `git config user.name 'First Last'` and `git config user.email 'name@domain.com'` to tell git what name and email to use in commit messages; you *must* run these commands before you run git for the first time.
- If using git 1.8 or later, I strongly suggest running `git config --global push.default simple` so `git push` behaves a bit more intuitively.

Basic git commands

Command	Effect
`git init .`	Create a new git repository in the current directory
`git clone <url> foo`	Copy the repository at `<url>` into directory `foo`. If dir. name `foo` is omitted, `clone` will generate one based on `<url>`.
`git add foo`	Stage changes to file `foo` (if existing) or add it to repository (if new). Must run `git commit` for changes to take effect.
`git rm foo`	"Remove" file `foo` from the repository. Must run `git commit` for changes to take effect.
`git mv foo bar`	Rename file from `foo` to `bar`. Must run `git commit` for changes to take effect.
`git commit`	Commit staged changes to repository
`git commit -a`	Stage all changes to tracked files and commit them to repository
`git status`	Show changes in repository relative to last commit
`git diff foo`	Show diff of unstaged changes in file `foo` made since last commit. If file name `foo` is omitted, show unstaged changes in all tracked files.
`git log -<N>`	Show messages for last `<N>` commits; if `-<N>` is omitted, show entire repository history.

Working with branches and remotes

Command	Effect
`git checkout -b foo`	Create a new local branch named `foo` and switch to it
`git checkout foo`	Switch to an existing local branch named `foo`
`git branch`	List all local branches; current branch indicated with an '*'.
`git branch -vv`	List local branches with last commit info and upstream branch (if any)
`git merge foo`	Merge changes from local branch `foo` into current branch.
`git remote add baz <url>`	Add `url` as a remote named `baz`
`git fetch baz`	Get changes from all branches in remote repository `baz`. **NOTE**: this does *not* actually merge remote changes into the current branch.
`git fetch baz bar`	Get changes from remote branch `bar` in remote repository `baz`. **NOTE**: this does *not* actually merge remote changes into the current branch.
`git checkout -t baz/bar`	Create a new local branch named `bar` with remote branch `baz/bar` as its upstream
`git merge baz/bar`	Merge changes from remote branch 'baz/bar' into the current local branch.
`git pull`	`fetch` and `merge` changes from remote upstream into current local branch.
`git push`	Send updates from current local branch to its remote upstream.
`git push baz bar`	Create (or update) remote branch 'baz/bar' with local HEAD (last commit).