# #1 Daily Tools

#advanced-git

## The `.gitignore` file

If there are any files you don't want to be tracked by Git, you can add it to the `.gitignore` file located in the base of your project.  The file might not already exist if you're creating a new project.

```
$ echo "*.pdf" > .gitignore # Ignore all pdfs
```

Simple as that. Now, Git will not show any pdfs under untracked files.

## Clean

If you find that your project suddenly has a bunch of untracked files that shouldn't be there, `git clean` will remove them for you. Obviously, this is a dangerous command. It's so dangerous in fact, that it won't even work by default:

```
$ git clean
fatal: clean.requireForce defaults to true and neither -i, -n, nor -f given;
refusing to clean
```

Supplying the `-f` flag will usually do what you want, which will remove all untracked files. Again, this is dangerous! Make sure you are ok with losing the files.

```
$ git clean -f
```

Note that `git clean` will only remove untracked files, so it's different than `git checkout -- filename`.

## Stashing

`git stash` will clean up your repository and remove all the untracked changes. This will make it appear like the current `HEAD`. However, it will also save those changes so you can look at them later.

To stash your changes:

```
$ git stash push # also git stash
```

To look at your stashed changes:

```
$ git stash list # Get a list of all your stashes
$ git stash show # Show your most recent stash
```

The `-p` flag can be applied to the examples above to see a diff of the changes.

To apply the changes (even on another branch!):

```
$ git stash apply
```

To remove stashed contents:

```
$ git stash drop # Removes most recent stash
```

As a shortcut:

```
$ git stash pop # Does git stash apply then git stash drop
```

## Tags

Tags are kind of like branches in that they are a way to label commits. Unlike branches, tags do not move when you create new commits.

```
$ git tag <tagname> # Create a new tag with a given name
$ git tag -d <tagname> # Delete a tag
```

Tags tend to be more permanent than branches. Try not to get into the habit of deleting old tags if possible, since this has possible consequences for tags on remotes.

## Fast-forward Merges

In certain cases, it might be unnecessary to create a merge commit. By default, Git will attempt to do a “fast forward” when possible.

```
$ git checkout -b feature # This will checkout to a new branch
$ echo ... # Edit files here
$ git commit # Commit on feature branch
$ git checkout master
$ git merge feature
```