In this lecture, we will discuss…

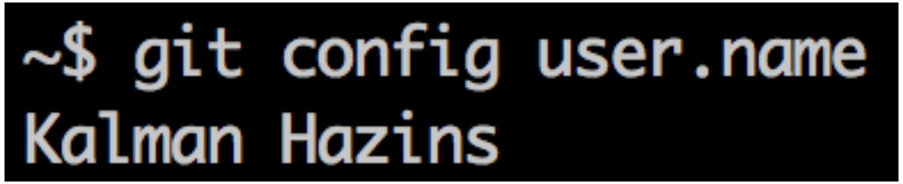
- ✧ Setting up your environment for Git
- ✧ Interacting with your local Git repo

Git Setup

- Setup properties **globally**
  - `$git config --global user.name “Kalman Hazins”`
  - `$git config --global user.email` my@example.com

Git Setup

- ✧ Verify that an option has been set
  - `$git config <option>`
  - For example, `$git config user.name`
- ✧ Getting help on any Git command
  - `$git help <command>`



```
~$ git config user.name
Kalman Hazins
```

Initializing a Repo

- Where do I get a repo from?
  1. Create a **new repo**
     - `$cd workding_dir`
     - `$git init`
     - (Possibly create a `.gitignore` file)
     - `$git add .`

       (. Adds the entire current directory with subdirectories)
     - `$git commit -m “Initial commit”`

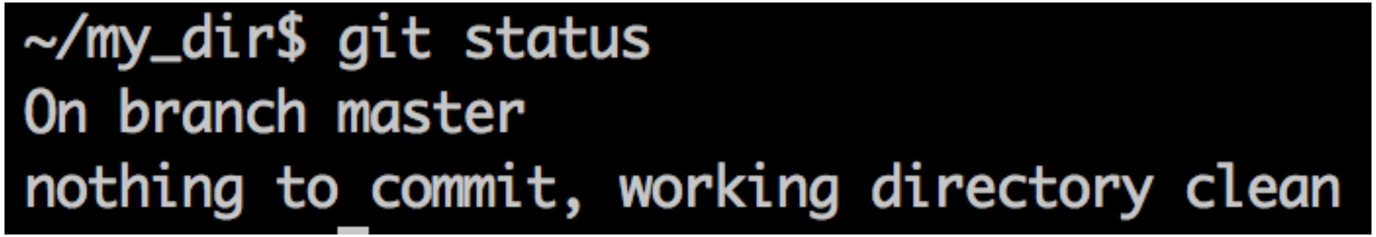
Cloning a Repository

2. **Clone** an existing repo (for example from Github)

- `$git clone https://repourl.git`
- Many **transfer protocols** available
  - https:
  - git:

git status

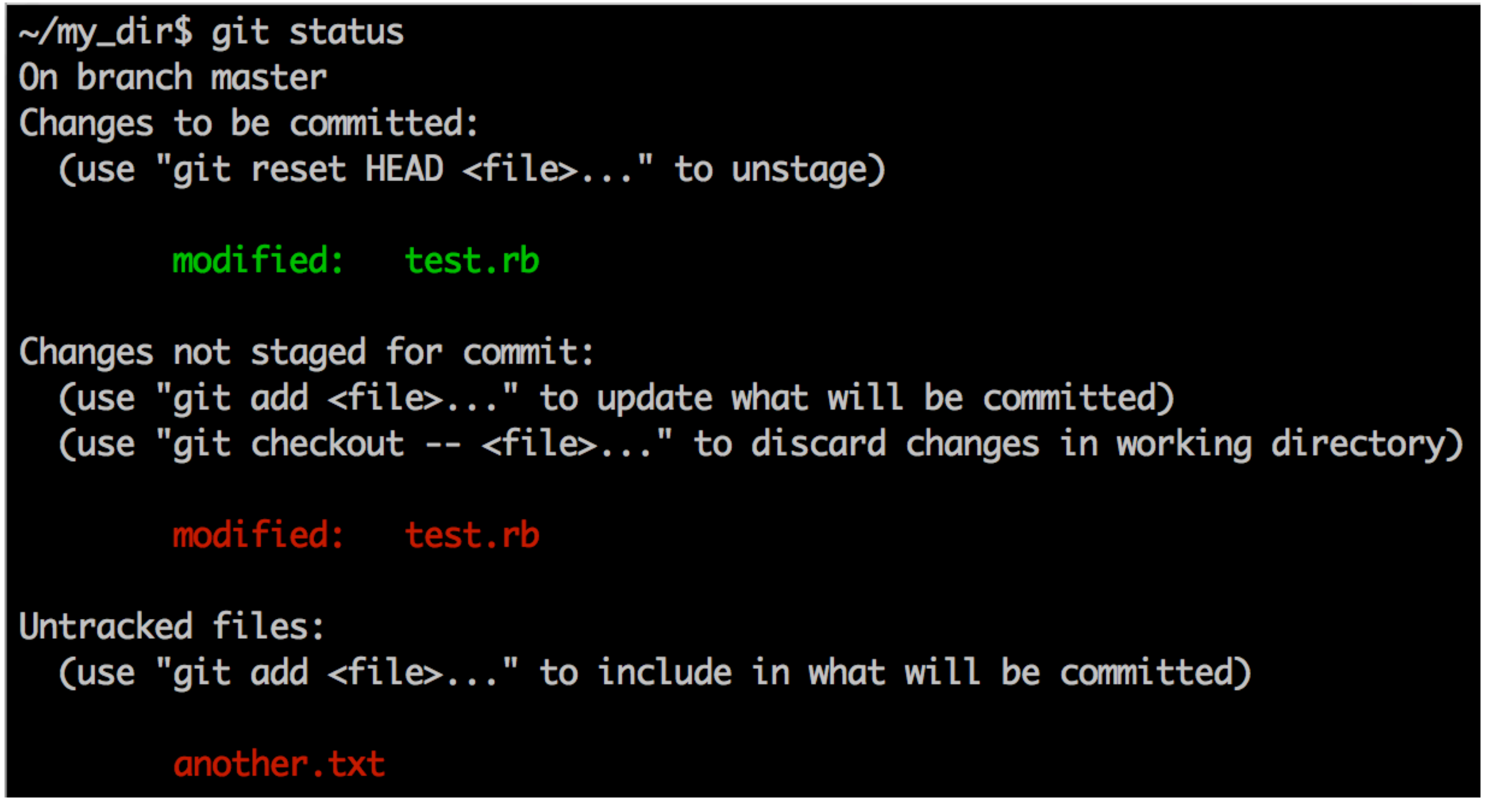
✧ `$git status`
- Provides the **current status** of your repo



```
~/my_dir$ git status
On branch master
nothing to commit, working directory clean
```

git add

- `$git add <file/dir>`
  - Add untracked file(s) to be tracked or
  - Add a modified tracked file to the staging area


```
~/my_dir$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   test.rb

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   test.rb

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        another.txt
```

Mods made to the file after “git add” – need to be “git-added” again – even if you did not commit yet

git diff

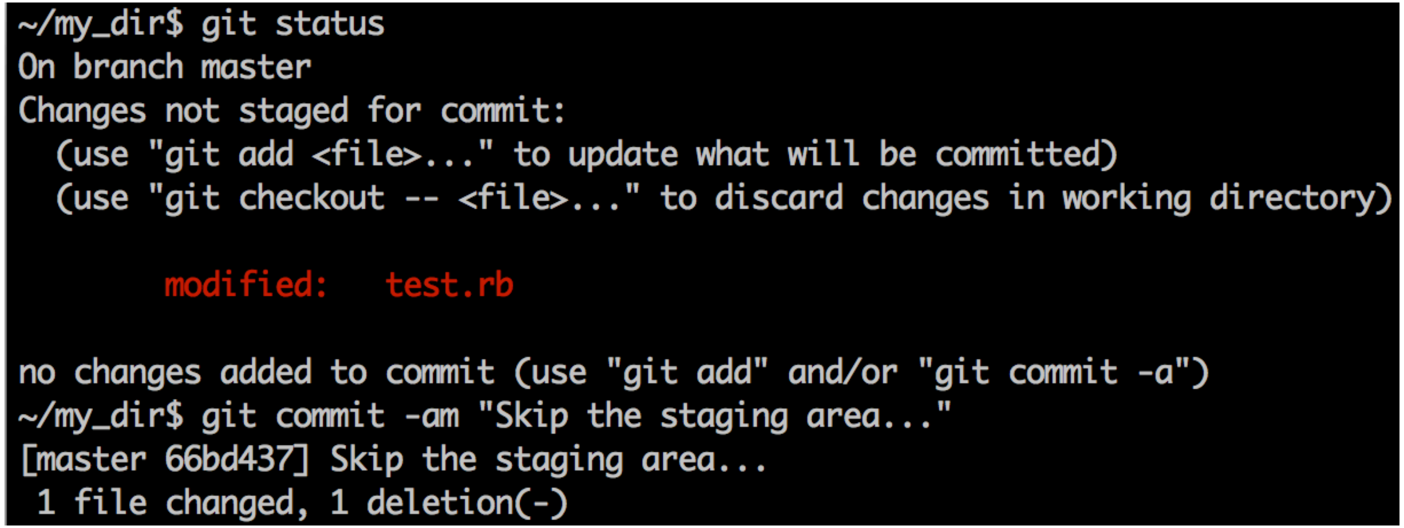
- ✧ `$git diff`
  - Shows the **difference** between staging and working directory
- ✧ `$git diff --staged`
  - Shows the **changes** between HEAD (latest commit on current branch) and staging directory
- ✧ `$git diff HEAD`
  - Shows the **deltas** between HEAD and working dir

git commit

- `$git commit`
  - Commits your changes to the repo
    - Prompts for a **commit message** in an editor
  - **Better**, just use the `-m` (message) option
    - `$git commit -m "Your msg here"`

Skipping the Staging Area

- To **skip** the staging area - just use `-a` flag
  - **After** initially adding the file!!!
- Either `-a` `-m` or `-am` will do the trick



```
~/my_dir$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   test.rb

no changes added to commit (use "git add" and/or "git commit -a")
~/my_dir$ git commit -am "Skip the staging area..."
[master 66bd437] Skip the staging area...
 1 file changed, 1 deletion(-)
```

Going Back in Time

- Before committing
  - `$git checkout .`
    - Re-checkout all tracked files **overwriting** local changes
  - `$git checkout -- <file>`
    - Re-checkout **just one** specific file
- After committing
  - `$git revert HEAD`
    - Reverts the **most recent** commit

Summary

- You have to add a file for tracking at least once before it can make it into the repo
- Can easily go “back in time” to a snapshot

What’s next?

- Github and remote repositories in Git