# Tips

* Keep Master Branch the one that “always works”
* Split into two branches for two people to do their own things, then merge later (merge back into master branch, for example)

# Basics

Repository: current file + histories

Clone: “make a copy of the repository”

git log

* Arrow to scroll, PgUp, PgDown
* Q to exit

git log --

$ git log | grep -A 4 "Revert controls"

Revert controls

Git diff <commit\_id1> <commit\_id2>

* With commit\_id: shows the difference between two commits

Git diff

* Compare files in working directory and staging area (i.e. previous “add”)

Git diff --staged

* Compare between files in staging area and repository (i.e. previous “commit”)

Git checkout <commit\_id>

* Check out a particular commit
* Eg. a bug was found since version XXX, but you can’t really remember which version. So you want to try to
* To create a new branch, do Git checkout –b <commit\_id>
  + Actually doing two commands:
    - git branch <branch-name> (create a new branch)
    - git checkout <new-branch-name> (switch to new branch. Further commits will be on that branch)

## Setting up a new repository

Go to the desired directory and

git init

* Note that it does not create any commits yet
* HEAD = current commit

Git status

* Status since last commit
* On master branch?

## Add to staging area

Git add <filename>

* Add them to the staging area (not yet repository)

## Commit from staging area to Repository

Git commit

* Staging area -> repository

# Branches

## Show current branch

Git branch

* Shows current branches
* \* means the branch that is currently checked out.

## Create a new branch

Git branch <new-branch-name>

* Create new branch with new branch name.

## Switch to new branch

Git checkout <branch-name>

* Switch to branch

## Merging branches

1. Check out the master branch
2. Git merge <master> <branch2>
3. Git branch –d <old branch> delete the label for branch2

## Showing commits relative to its parent

Git show <commit-id>

1. Useful when after merge, want to compare changes with its direct parents

## Terminologies

* **Remote branch**: a branch created by someone else

# GitHub

## Add a repository on Github as remote

Git remote add <name you want to use> <URL at Github>

Git remote –v (for verbose)

1. See which are the remote repositories

## Send Changes to the Remote using Push

Git push <remote> <branch you wish to push>