Github.com/new – to create a new repository

1. **Git clone:** need to have git installed, type: git clone <url>; downloads a copy of all contents on the server repository.
2. **Touch:** touch hello.html; create the file hello.html, touch is a terminal command for creating a file.[Doesn’t work in windows so either use nul > hello.html or add a file touch.cmd in path with code:

@echo off

setlocal enableextensions disabledelayedexpansion

(for %%a in (%\*) do if exist "%%~a" (

pushd "%%~dpa" && ( copy /b "%%~nxa"+,, & popd )

) else (

type nul > "%%~fa"

)) >nul 2>&1

**Commit: take a “screenshot” of the current state.**

1. **git add**: git add <filename> ; 🡨 tracks this file when we commit(save).
2. **git commit**: git commit –m “message”; creates a save point with changes to tracked files[local save]
3. **git status**: what is happening in current repository, e.g. my branch is ahead of origin/master branch by 1 commit.[means my local copy has 1 changes compared to server file]
4. **open:** terminal command to open a file; in windows use **start** command, e.g. start hello.html will open hello.html in default browser.
5. **Git commit –am “message”:** Adds all files and commits with the message.
6. **Git pull:** download the latest version from the server to your local repository.
7. **Editing files on github:** can also edit on github then commit using the button and entering a message.
8. **Merge conflicts:** sometimes u need to fix conflicts before u can commit

**<<<< HEAD**

B=0; 🡨 **Your changes**

=======

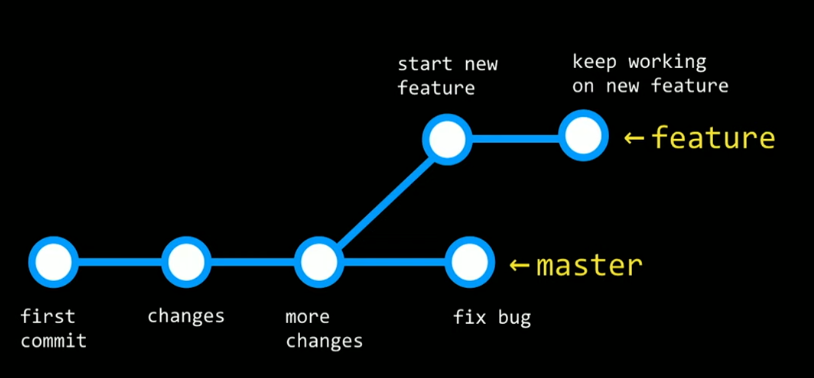
B=2; 🡨**Remote changes**

**>>>> 59a05f23d034baff124** 🡨 Hash of the conflicting commit

Remove all blank lines and decide which version to keep then commit changes.

1. **git log:** lists all changes with their commit hashes and commit messages. Also who made the changes and when.
2. **git reset:** git reset --hard <commithash> ; go back to the commit hash. git reset --hard origin/master; go back to the version on github/server.

**Branching: working on different parts of the repo at the same time.**



1. **head:** points to a branch, to work on a branch first head must point to it.
2. **git branch:** tells the current branch (indicated by the \*).
3. **git checkout:** git checkout –b newbranchname; -b for new branch otherwise just type branch name, e.g. git checkout master.
4. **git merge:** to merge branches. E.g. git merge style will merge current branch with ‘style’ branch

**Fork: create your own copy in your repository of a github repo.**

**Github pages: free static website, yourusername.github.io**

Go to github.com/new, create a new repository with the name: yourusername.github.io

Clone the repo, create index.html, push changes

Go to settings you can see “your site is ready to be published at yourusername.github.io”