Git

Stages of a file

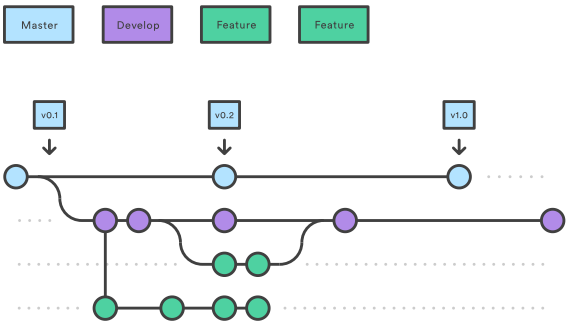
A file in a git repository can be

- **untracked**: not tracked
- **not modified**: file has been commited and not changed since then
- **modified**: file has been changed
- **staged/added**: file will be include in the next commit

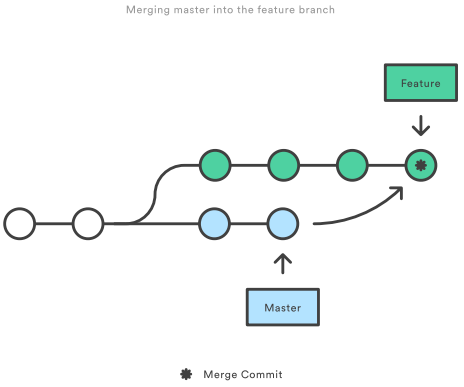
Key Terms and Commands

- **Repository**: A place that is being kept track of
- **Init**
  - initializes a repository on your local machine (not online!)
  - `git init`
  - a .git folder will be created
- **Add**
  - adding files to be commited
  - `git add <file>`
  - `git add -u`: Add all files that were include previously
  - `git add --all`: All files
- **Commit**
  - make changes permanent
  - Push: publish (local) commits to remote repository
  - Fetch (pull): retrieve commits from remote repository
  - `git commit -m “message”`

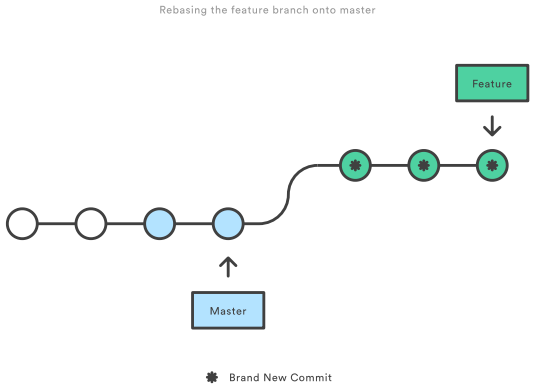
- **Tag**
  - add tags to the head (current position)
  - `git tag <Name>`
  - e.g: `git tag "Version 1.1`"
- **Checkout**
  - `<Tag>, <commit(ID)>, <branch>`
  - you can checkout (visit) tags, branches and commits
- **Branches**
  - Versions/Timelines of documents
  - “branch early and branch often”
  - `git checkout -b <name>` to create new branch and switch to it
  - or `git switch` to switch between branch and main



- **Merge**
  - Combine changes of one branch into another branch
  - keeping parallel history
  - `git merge <branch>`
  - if files cannot be merged, you’ll get an error and the file will have merge markers that indicate where the conflict occured



- **Rebase**
  - like merge but it makes it look like the changes were made sequentially and not parallel
  - needs to be on branch that going to be rebased into main
  - `git rebase main`



- **Fork**
  - forking a repository online creates a duplicate and makes you the owner of the duplicate
- **Clone**
  - downloads the repository to your local machine
- **Fetch**
  - **downloads** newer commits from the remote
    - however you might need to merge the “online” changes to see the new files in your finder/explorer
  - the detached head will go onto the remote head and the local head will stay
    - a detached head is the head of the remote repo
  - `git fetch`
- **Pull**
  - combines the commands **fetch** and **merge** to o/main
    - classic way of updating your local machine
  - `git pull`
  - `git pull --rebase` : combines **fetch** and **rebase**
- **Push**
  - push local changes to remote (requires permission and password)
  - `git push`
  - `git push --rebase`

- **Stash**
  - save intermediate changes without a commit
  - `git stash`
- **Reset**
  - to abort changes and return to last commit
- **Cherry Pick**
  - only merge one single commit into current branch

Useful commands

- git status: to see the file changes and some more useful info
- git diff: compare head and new changes
- git log
  - q for exit
- Git Workflows

  > Git Workflow | Atlassian Git Tutorial
  >
  > Git is the most commonly used version control system today. A Git workflow is a recipe or recommendation for how to use Git to accomplish work in a consistent and productive manner. Git workflows encourage developers and DevOps teams to leverage Git effectively and consistently.
  >
  > https://www.atlassian.com/git/tutorials/comparing-workflows

- git Extension in VSCode: mhutchie.git-graph