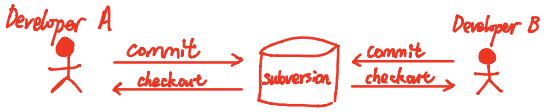
Github

- pull request: After changing code locally, make a request to merge on github.
  - check differences
  - comment on specific code
- watch: Add repository to "watch", you can check "news feed" (updates, bug fixes, etc.)
- issue: Assign an "issue #" to every pull request, every issue corresponds to certain lines of code.

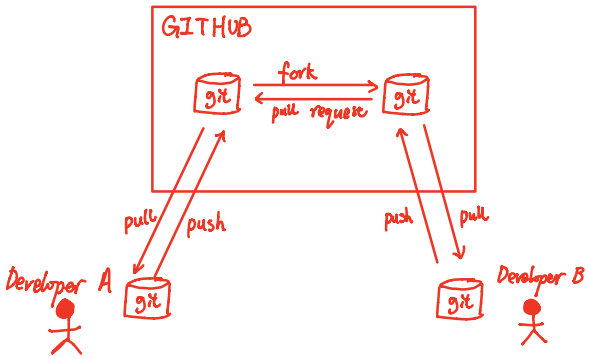
Git

- What is Git? Version Control.
  - Developed by Linus Torvalds
  - Manage update history across versions
- Git vs. Subversion

subversion:



git:



- Benefit of Git:
  1. It is a distribution system. User can fork repository & edit.
  2. No need to connect to internet. Work locally.

- Git installation
  - Mac & Linux: pre-install.
  - Windows: online package.
    - LF vs CRLF
      LF: Mac/Linux line ending style
      CRLF: Windows line ending style
- Git usage
  - Link git to Github
    1. setup name & email (will be shown on update log, *public)

       git config --global user.name "Firstname Lastname"

       git config --global user.email "Email"

    2. Increase Readability (change color.ui)

       git config --global color.ui auto

    3. Setup SSH key

       in Git:

       1. create a new key

          ssh-keygen -t rsa -C "email" ← save location (default), password twice

       2. inside default folder, two keys are created: public & private

          id_rsa: private key id_rsa.pub: public key

       3. find public key

          cat ~/.ssh/id_rsa.pub

    4. add SSH key to Github

       - Go to SSH setting in Github, add title (Lowercase) and pasted key

    5. Link git to Github

       ssh -T git@github.com ← Yes, Password

Github usage

- Create Repository
  - README ← if there exists a git repository, unclick
    if no existing repository click
  - GFM (Github Flavored Markdown)

- Clone Repository (Download repository from github to Local)

  git clone ____SSH or HTTP____ *use SSH needs protection key. find them in github repository.

- Update Repository (Push updated file from local to github)

  1. add a file into the folder (local repository)
  2. git status (show repository status)
  3. git add "Filename" (put files into staging area)
  4. git commit -m "message" (capture snapshot of current staged changes)
  5. git push (push all files)

  - git log (check submission log files)

Basic git commands

- git init (initial new repository)
  - use in a folder / directory
  - create a .git file that tracks everything in the directory
- git status (check repository status)
  - if modify repository or working tree, git status changes

- git add "Filename" (put files into staging area)
- git commit -m "message" (capture snapshot of current staged changes)
  - can recover files in working tree
  - more detailed description, please use git commit
    - Press 'i' to edit, Press 'esc' to abort
    - Press 'shift+z' twice to exit / save changes
    - '#' for comment, leave empty abort changes
- git log (check submission log files)
  - show first line git log --pretty=short
  - git log "filename" (log specific files)
- git diff (difference between working tree & staging area or latest submission)

  \+ : New

  \- : Remove

  - before adding to staging area, git diff will compare working tree with latest submission
  - after adding to staging area, working tree = staging area, use git diff HEAD

  * Good habit: use git diff HEAD before submission

Branch

- git branch (list all branches)
  - '*' means current branch
- git checkout -b "name" (create new branch and switch to new branch)

  or

  git branch "name" + git checkout "name"

  (create new branch) (switch to new branch)

- inside branch, you can edit all you want, it doesn't affect main branch
- git checkout "name" (switch to existing branch)
- git merge --no-ff "name"
  - --no-ff means record merging action in log
  - do this in 'main' branch
- git push (submit code)
- git log --graph (check branch in graphics mode)

- git reset --hard "HASH Value" (go back to certain snapshot)
- git reflog (check all logs executed in the current repository) ← can use again
- git commit --amend (edit commit message)
- git rebase -i HEAD~2 (edit history / log within 2 changes)

  git branch -d "Branch Name" (delete local branch)

Remote repository

- initialize repository on github without ReadME
  - git remote add origin "SSH or HTTP"
  - git push -u origin main

  * everything is the same
- branch manipulation
  - git checkout -b "branch name" (create local branch)
  - git push -u origin "branch name" (push local branch to remote branch)
  - git branch -a (list all branches)
  - git checkout -b "branch name" origin/"branch name" (switch to remote branch)

    → create a local branch
- git push (submit modified branch)
- git pull (download Latest update from remote repository to local)
- git push origin --delete "Branch Name" (delete remote branch)

Github functionality

- keyboard shortcut: Shift + /
- gist: snippets, reuseable code
- fork: copy other people's repository to yours.
- compare branch differences: url: https://github.com/mayichong/git-tutorial/compare/"A"..."B"

  A B

- compare branches between time interval: ... compare/master@{7.day.ago}...master

  day week month year

  or

  {year-month-day}

- issue # ⇔ pull request #
- get a diff or patch file from pull request: .../mayichong/"Repository name"/pull/28.diff ↕ 28.patch

Pull Request:

Definition: after changing source code, ask permission to merge code

User Scenario: when A uses a software created by B on github,

A found and fixed the BUG,

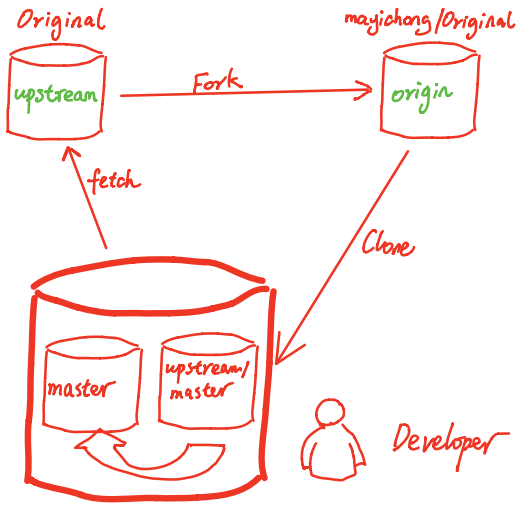
A can do a pull request on B's repository.

Steps to perform a pull request:

1. Fork + Clone Repository
2. Create a new branch
3. Switch to the new branch
4. Edit the file
5. Add + Commit changed files
6. upload local branch to remote repository
   - git push origin "Branch Name"
7. go on github, click "new pull request"
8. compare code differences, submit request
9. if branch is not done, write [WIP].

Repository Maintainence

- Problem: Fork / clone repository won't update by itself. What to do ???



Steps to constantly update repository

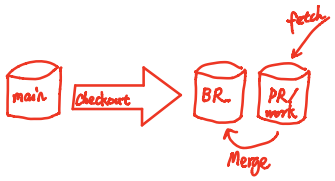
1. Fork
2. Clone
3. Add upstream to ORIGINAL repository, make it remote repository.
   - git remote add upstream "http or SSH"
   * setup once!
4. Fetch latest data
   - git fetch upstream (fetch data & merge with master)

Accept pull request

* Before merging, download locally and check if code works!
- You can send feedback specify lines of code

Steps to accept a pull request:

1. Clone/update repository locally
2. make the repository a remote repository from Pull requester
   - git remote add PRequester "Sender SSH or HTTP"
   - git fetch PRequester
3. Create a new branch (BR)
4. Merge PR with BR
   - git merge PRsender/work



5. if everything works, delete new branch.
6. Merge pull request into main
   1. Merge on github website
   2. Merge using git bash
      - git checkout main
      - git merge PRsender/work
7. check diff between local & remote repository
8. push! - git push

Hub

- Hub is a tool that extends git
- Hub can send request through github API

Hub commands:

- hub clone Hello-World (clone repository without the prefix)
- hub remote add (remote add, without the prefix)
- Rest of them are similar

Travis CI: use to test & deploy code after uploading to github

* Link to github for auto-testing

Coveralls: use to see cover ratio for code. Check if codes are not used during testing.

Jenkins: open source automation server.

test if pull request can damage original code or not