

getting started with git

김덕홍 연구원

2015-04-03

N A V E R | L | A | B | S |



git

원격 저장소

git remote repository

- 팀 개발 시 다른 사람과 작업 내용을 공유하기 위해 필요
- 개인 프로젝트시 백업 용으로 이용되기도 함
- github 등의 서비스를 이용하여 소스코드 공개용으로도 사용
- 일반적 remote repository 는 bare mode 로 만들어짐

git remote command

원격 저장소 추가

```
$ git remote add (remote alias) (remote url)
```

원격 저장소 상세정보 확인

```
$ git remote show (remote alias)
```

원격 저장소 alias 변경

```
$ git remote rename (old_name) (new_name)
```

원격 저장소 url 변경

```
$ git remote set-url (remote alias) (new remote url)
```

원격 저장소 목록 확인

```
$ git remote -v
```

git remote repository 등록하기 실습

```
~/git-repo $ git remote add origin https://github.com/insanehong/toolcon.git
```

```
~/git-repo $ git remote -v
```

```
origin      https://github.com/insanehong/toolcon.git (fetch)
```

```
origin      https://github.com/insanehong/toolcon.git (push)
```

```
~/git-repo $ git remote show origin
```

```
* remote origin
```

```
Fetch URL: https://github.com/insanehong/toolcon.git
```

```
Push URL: https://github.com/insanehong/toolcon.git
```

```
HEAD branch: (unknown)
```



git

원격 branch 다루기

git push command

local branch push 하기

```
$ git push (remote) (local branch_name):(remote branch_name)
```

```
$ git push origin master:mater
```

```
$ git push origin master:dev
```

동일 한 이름의 branch 로 push 하는 경우 local branch 생략

```
$ git push (remote) (remote branch_name)
```

```
$ git push origin mater
```

일반적으로 많이 사용



git push 실습

```
~/git-repo $ git push origin master
```

```
Counting objects: 23, done.
```

```
Delta compression using up to 4 threads.
```

```
Compressing objects: 100% (21/21), done.
```

```
Writing objects: 100% (23/23), 2.77 KiB | 0 bytes/s, done.
```

```
Total 23 (delta 8), reused 0 (delta 0)
```

```
To https://github.com/insanehong/toolcon.git
```

```
* [new branch]      master -> master
```

```
~/git-repo $ git push origin master:dev
```

```
Total 0 (delta 0), reused 0 (delta 0)
```

```
To https://github.com/insanehong/toolcon.git
```

```
* [new branch]      master -> dev
```


git push command

remote branch 강제로 덮어쓰기

non-fast-forward 일때는 push 가 거부 됨

```
$ git push -f (remote) (local branch_name):(remote branch_name)
```

```
$ git push -f origin todo:dev
```

remote branch 삭제 하기

```
$ git push (remote) :(remote branch_name)
```

```
$ git push origin :dev
```

git push 실습

```
~/git-repo $ git push origin todo:dev
```

```
Username for 'https://github.com': insanehong
```

```
Password for 'https://insanehong@github.com':
```

```
To https://github.com/insanehong/toolcon.git
```

```
! [rejected]      todo -> dev (non-fast-forward)
```

```
error: failed to push some refs to 'https://github.com/insanehong/toolcon.git'
```

```
hint: Updates were rejected because a pushed branch tip is behind its remote
```

```
hint: counterpart. Check out this branch and integrate the remote changes
```

```
hint: (e.g. 'git pull ...') before pushing again.
```

```
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

```
~/git-repo $ git push -f origin todo:dev
```

```
Total 0 (delta 0), reused 0 (delta 0)
```

```
To https://github.com/insanehong/toolcon.git
```

```
+ 548b398...b2e7fd5 todo -> dev (forced update)
```

```
~/git-repo $ git push origin:dev
```

```
To https://github.com/insanehong/toolcon.git
```

```
- [deleted]      dev
```



git

원격 저장소 clone

git clone

- 이미 프로젝트가 진행되고 있어 remote repository 가 존재 하는 경우
- 새로운 개발장비에 개발 환경을 세팅 해야하는 경우
- 현재 작업중인 directory 외에 프로젝트를 복사 하고 싶은 경우 등등

git clone command

remote 저장소를 local 에 clone 하기

```
$ git clone (remote_url) [directory name] -b [branch name]
```

생략 시 remote repository 명으로 만들어 짐

생략 시 remote repository
default branch 를 받아옴



git clone 실습

```
~/git-repo $ cd ~/git-workspace
```

```
~/git-repo $ git clone https://github.com/insanehong/toolcon.git
```

```
Cloning into 'toolcon'...
```

```
remote: Counting objects: 23, done.
```

```
remote: Compressing objects: 100% (13/13), done.
```

```
remote: Total 23 (delta 8), reused 23 (delta 8), pack-reused 0
```

```
Unpacking objects: 100% (23/23), done.
```

```
Checking connectivity... done.
```

```
~/git-repo $ cd toolcon
```

```
~/git-repo $ git lg
```

```
~/git-repo $ git remote show origin
```



git

원격 저장소
변경내용 받아오기

remote 변경 사항 받아오기

fetch

remote 저장소에 추가된 object 들을 받아와서 local 에 저장

```
$ git fetch (remote) (remote branch)
```

```
$ git fetch origin master
```

```
$ git fetch origin
```

pull

remote 저장소에 추가된 object 들을 local 로 받아오면서 merge 까지 수행

fetch + merge

```
$ git pull (remote) (remote branch):(local branch)
```

```
$ git pull origin master:master
```

```
$ git pull origin master #현재 작업중인 branch 에 merge
```

```
$ git pull
```


remote 변경 사항 받아오기 실습

- toolcon directory 에서 발표자료 업데이트 후 remote 저장소로 push
- git-repo directory 에 toolcon directory 에서 변경한 내용 반영

remote 변경 사항 받아오기 실습

```
~/git-repo $ cd ~/git-workspace/helloworld
```

```
~/git-repo $ vi index.html //sample3 의 index.html 복사
```

```
~/git-repo $ git add index.html
```

```
~/git-repo $ git commit "update lecture note"
```

```
~/git-repo $ git lg
```

```
~/git-repo $ git push origin master
```

```
~/git-repo $ cd ~/git-workspace/git-repo
```

```
~/git-repo $ git lg
```

```
~/git-repo $ git pull origin master
```

```
~/git-repo $ git lg
```



git

**non-fast-forward
push 거부 해결하기**

non-fast-forward push 거부가 발생하는 경우

- remote branch 에 새로운 커밋이 추가 된 경우
- git push -f 옵션으로 강제로 remote branch 가 변경 된 경우 등등

non-fast-forward push 거부가 발생하는 경우

```
~/git-repo $ git push origin master
```

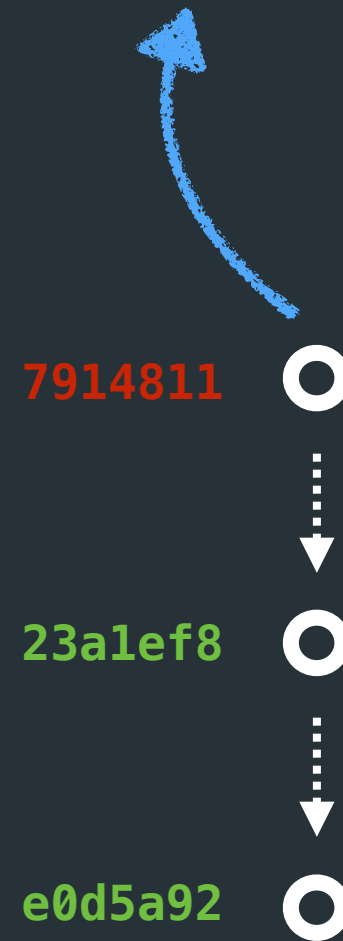
remote repository

다른 사람이 push 한 commit



local repository

local 에서 작업한 commit



non-fast-forward push 거부 해결 - merge

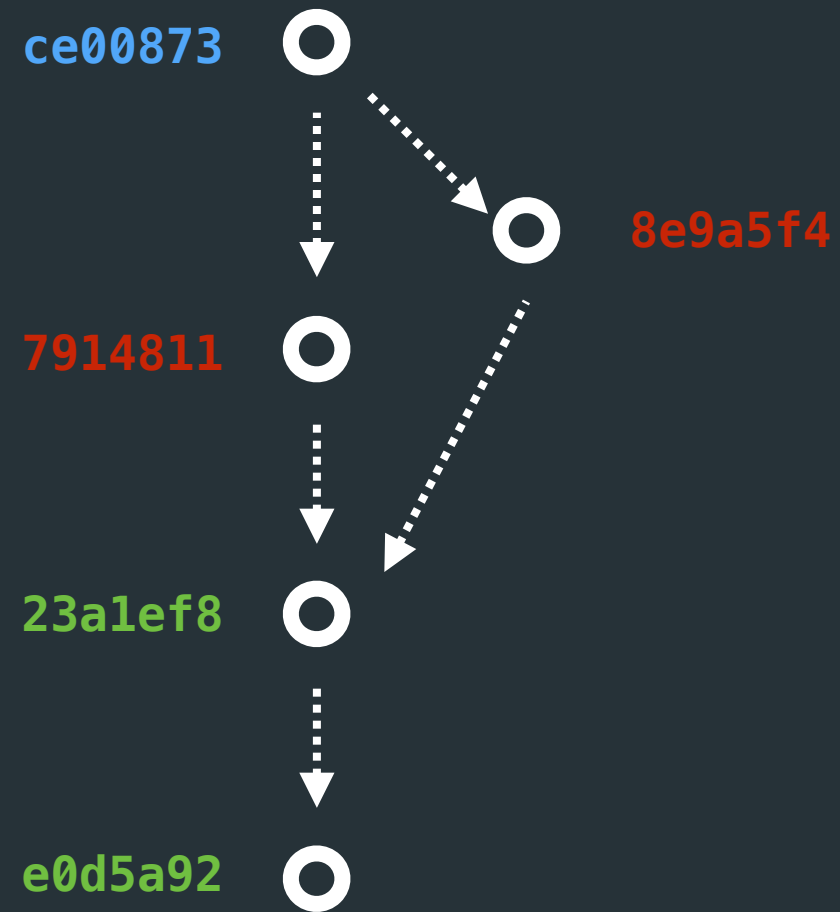
```
~/git-repo $ git pull origin master
```

remote repository

다른 사람이 push 한 commit



local repository



non-fast-forward push 거부 해결 - merge

~/git-repo \$ git push origin master

remote repository



local repository



non-fast-forward push 거부 해결 - rebase

```
~/git-repo $ git fetch origin master
```

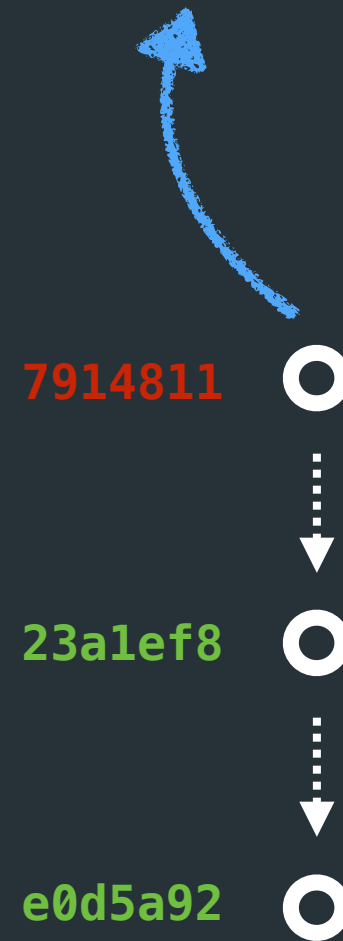
remote repository

다른 사람이 push 한 commit



local repository

local 에서 작업한 commit



non-fast-forward push 거부 해결 - rebase

```
~/git-repo $ git rebase origin/master
```

remote repository



local repository



non-fast-forward push 거부 해결 - rebase

```
~/git-repo $ git push origin master
```

remote repository

ad33csd ○



8e9a5f4 ○



23a1ef8 ○



e0d5a92 ○

local repository

ad33csd ○



8e9a5f4 ○



23a1ef8 ○



e0d5a92 ○



git

commands

git 유용한 commands

git 을 사용하면서 일어났던 과거 이력을 조회

```
$ git reflog
```

git 을 사용한 debugging

```
$ git bisect
```

커밋을 자유롭게 수정 병합 삭제 하기

```
$ git rebase -i
```

작업 중이던 내용 임시저장 하기

```
$ git stash
```

다른 branch commit 가져오기

```
$ git cherry-pick
```



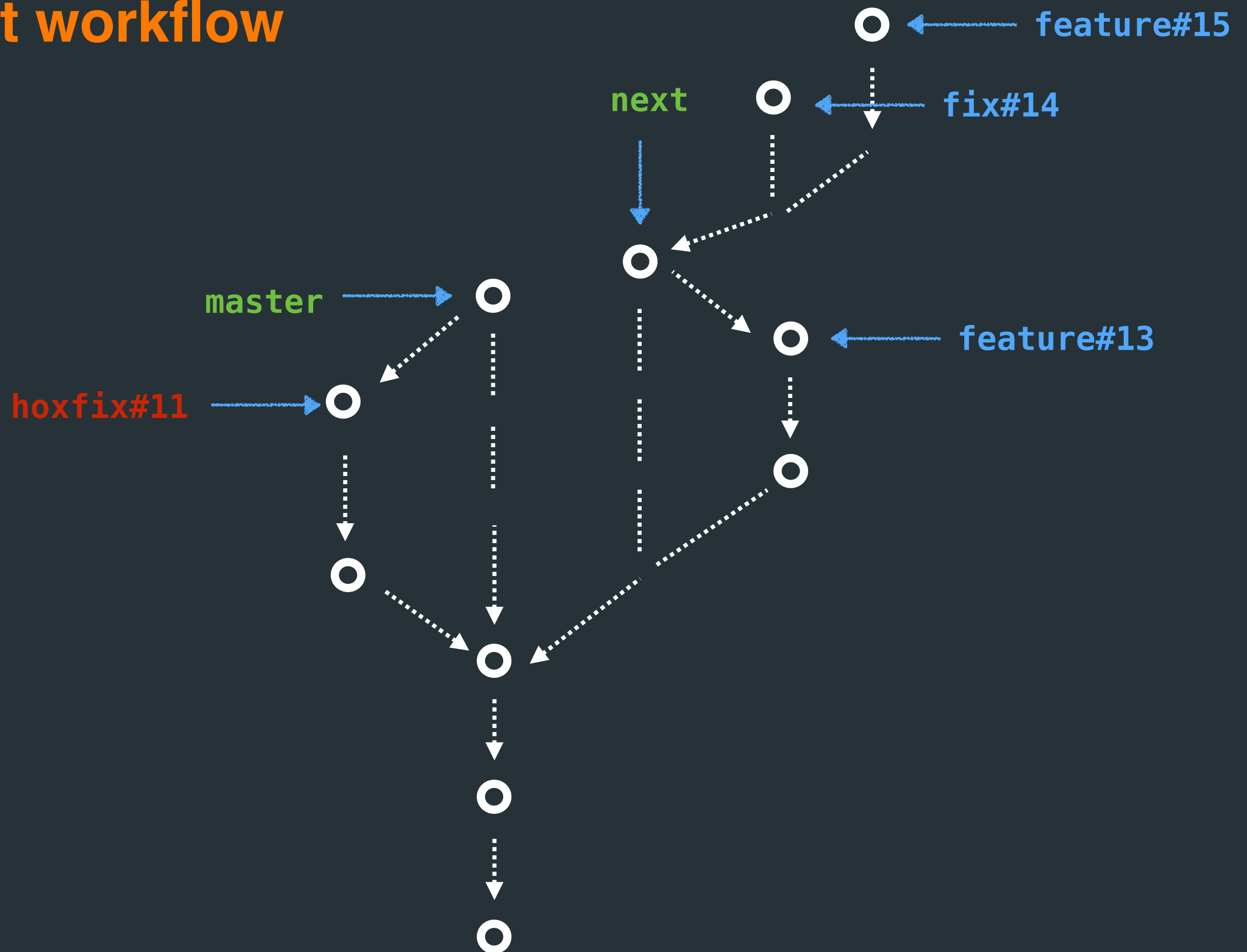
git

workflow

git workflow

- branch 전략과 맞물린 프로젝트 진행 방법
- branch model 과 fork model 중 선택 하는게 우선
- branch 전략을 사용하게 된다면 branch 관리를 어떻게 할것인가를 결정해야함
- 가장 좋은 방법은 topic == branch

git workflow

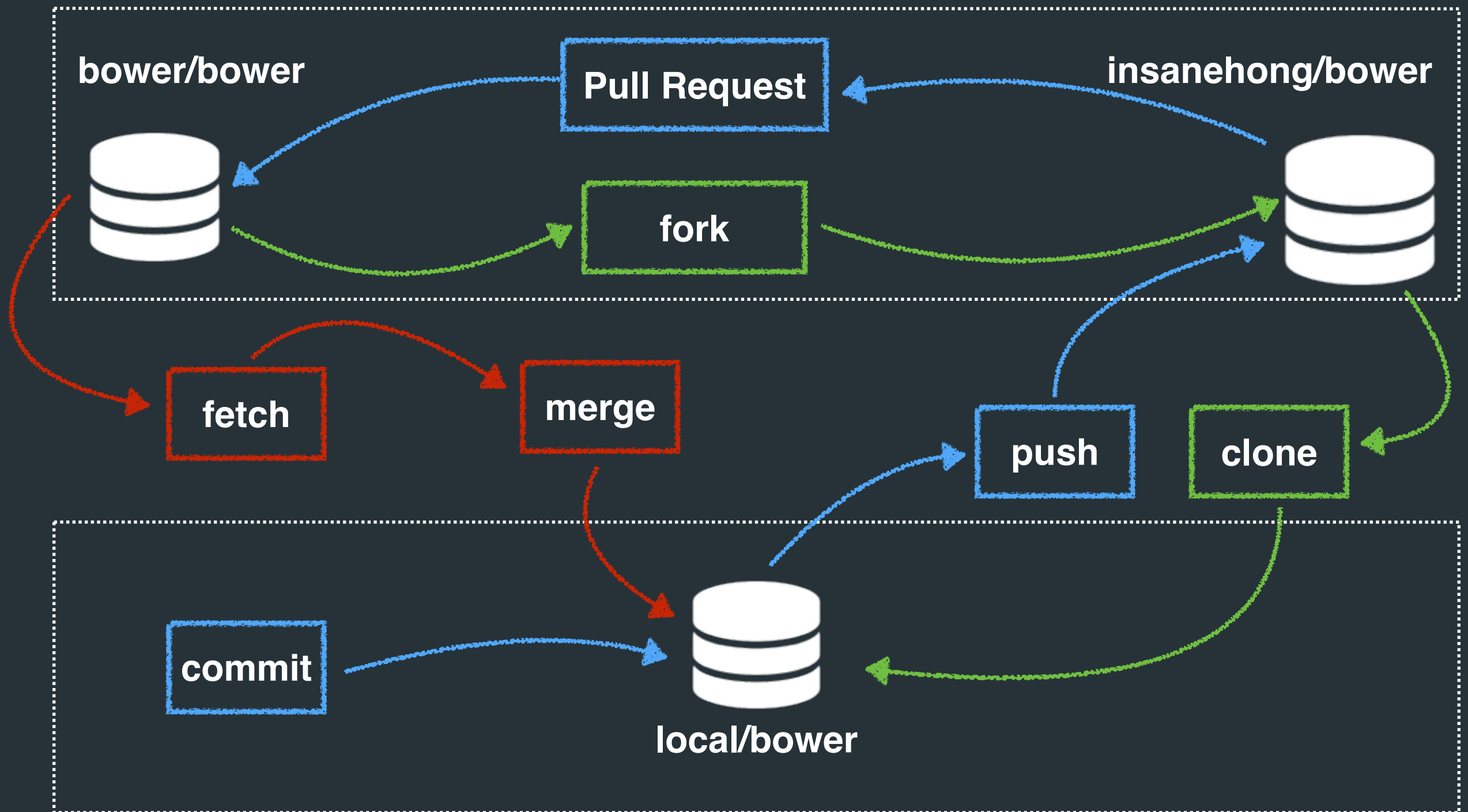




git

Open source project

open source contribution workflow



open source contribution

- 반드시 CONTRIBUTING.md 를 먼저 읽기
- 이슈/버그 트래킹을 하는 경우 중복 이슈가 있는 지 검색 해볼 것
- code review 에 당황하지 말고 잘 모르겠으면 수정 방향을 질문해 볼 것
- 그냥 한번 해볼 것(오타를 찾아 수정하는 것도 가장 간단한 기여방법중 한가지)


어려운만큼 알고나면 엄청난 효과

- 프로젝트 진행 중 발생하는 코드 관리 이슈의 90% 이상이 더이상 문제가 되지 않음
 - 익숙해지면 고민하기 전에 해결 방법이 머리속에 그려지게 됨
 - 더 익숙해지면 commit 을 마음대로 가지고 노는 본인의 모습을 보게 됨
- 어느 순간 쓰기 불편했던 오픈소스를 수정하여 internal patch 가 아닌 문제를 해결하는 patch 를 origin project 에 보내는 모습을 보게 됨

다른 것 몰라도 이것만이라도

- Pro Git 2장, 3장, 9장
 - <http://git-scm.com/book/ko/v1>
- 잘 안되더라도 직접 해보면서 몸으로 익히기

Q & A

A black and white photograph showing the back of a person wearing a dark-colored t-shirt. The t-shirt has the text "Everybody needs a hacker" printed on it in a white, sans-serif font. The person's hair is visible at the top of the frame, and the background is blurred.

Everybody needs a hacker

```
insanehong.talk.end(“Thanks. Bye!!”)
```

Reference

- Progit : <http://git-scm.com/book/ko/v1>
- Git Official Document : <https://www.kernel.org/pub/software/scm/git/>

본 자료는 크리에이티브 커먼즈 저작자표시-비영리-변경금지(CC BY-NC-ND) 3.0 Unported 라이선스에 따라 이용할 수 있습니다.

본 자료에 사용된 이미지들은 Creative Common License 를 따르며 이미지 출처는 해당 이미지 하단에 기재되어 있습니다.

twitter : @insanehong
email : insane.hong@navercorp.com

