

# GIT Cheat Sheet

## - add

파일의 변경사항을 다음 커밋에 반영하기

현재 WD의 파일을 SA로 이동

```
$ git add [file]
```

모든 변경 사항을 다음 커밋에 반영하기

```
$ git add . / $ git add -all
```

## - show

현재 브랜치의 가장 최근 커밋 정보 확인하기

```
$ git show
```

특정 커밋 정보 확인하기

```
$ git show [commit ID]
```

특정 브랜치의 가장 최근 커밋 정보 확인하기

```
$ git show [branch]
```

## - commit

SA의 파일을 메시지와 함께 커밋하기

```
$ git commit -m "commit message"
```

add하고 commit 동시에 하기

```
$ git commit -am "commit message"
```

마지막 커밋 수정하기

```
$ git commit --amend
```

## - clone

기존에 사용 중인 저장소를 clone해서 가져오기

```
$ git clone [url]
```

```
$ git clone [url] [새로운 폴더명]
```

## - push

로컬 저장소의 커밋한 변경사항을 원격 저장소로 전송하기

```
$ git push
```

```
$ git push [remote] [branch]
```

## - log

commit history 확인 (마지막 커밋인 head부터 이전 모든 이력 로그 표시)

```
$ git log
```

한 줄로 commit history 보기

```
$ git log --oneline
```

한 줄, 그래프 형태로 commit history 보기

```
$ git log --oneline --graph
```

```
$ git log --oneline --graph --all
```

commit의 변경사항까지 확인하기

```
$ git log -p [file]
```

## - pull

원격 저장소의 변경사항을 로컬 저장소로 가져오고 병합하기

```
$ git pull
```

```
$ git pull [remote] [branch]
```

## - fetch/merge

원격 저장소의 변경사항을 로컬 저장소로 가져오기 전 변경내용 확인하기

**\$ git fetch**

원격 브랜치를 현재 사용 중인 로컬 브랜치와 병합하기

**\$ git merge [remote] [branch]**

## - switch

브랜치로 이동하기

**\$ git switch [branch]**

브랜치 생성과 동시에 이동하기

**\$ git switch -c [new-branch]**

이전 브랜치로 전환, 이동하기

**\$ git switch -**

## - branch

브랜치 목록 조회하기, \*가 현재 브랜치

**\$ git branch**

전체 브랜치 목록 조회

**\$ git branch -a**

브랜치 생성하기

**\$ git branch [new-branch]**

브랜치 삭제하기

**\$ git branch -d [branch]**

브랜치 강제 삭제하기

**\$ git branch -D [branch]**

## - revert

원하는 커밋으로 되돌아가기 (커밋 이력에 되돌리기 커밋 추가)

**\$ git revert [commit ID]**

## - checkout

브랜치 이동하기

**\$ git checkout [branch]**

브랜치 생성과 동시에 이동하기

**\$ git checkout -b [new-branch]**

이전 브랜치로 전환, 이동하기

**\$ git checkout -**

## - reset

reset 이후에 커밋 내용만 수정, WD와 SA는 이전과 동일

**\$ git reset --soft [commit ID]**

reset 이후에 커밋 내용과 SA가 동일

**\$ git reset --mixed [commit ID]**

reset 이후에 커밋 내용, WD, SA 모두 동일

**\$ git reset --hard [commit ID]**

reset 이후에 다시 이전 상태로 돌아가기

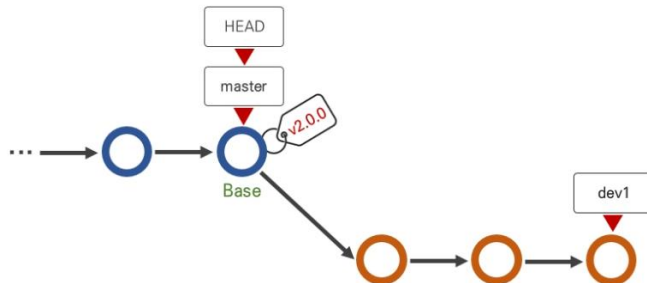
**\$ git reset --hard ORIG\_HEAD**

## - fast forward merge

merge할 대상이 현재 커밋의 직접적인 뿌리가 되는 경우

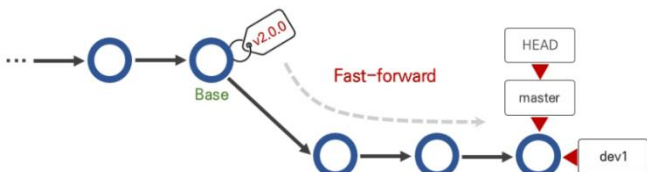
두 브랜치가 공통으로 가지고 있는 커밋

-> **base**



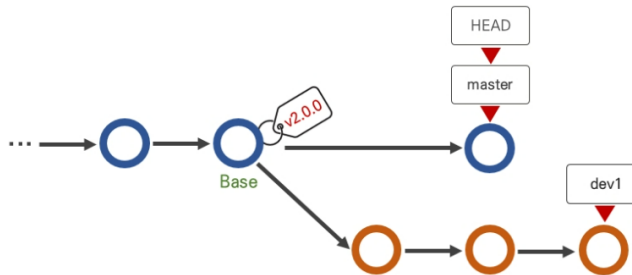
**\$ git merge dev1**

-> 새로운 커밋이 생기지 않는다



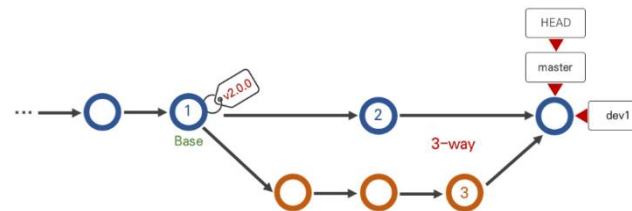
## - 3-way merge

두 브랜치 모두 **base**에서 커밋을 진행해서 분기해 나간 상태



내용을 병합할 때 **base**와 각 브랜치 2개가 참조하는 커밋을 기준으로 병합한다

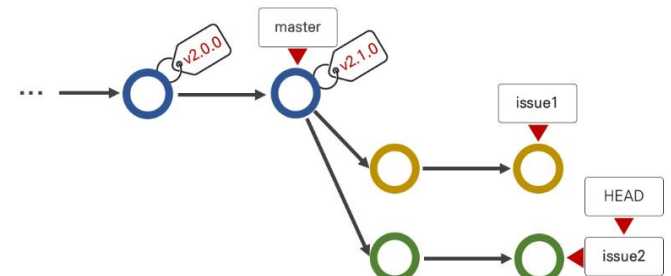
git merge를 하면 새로운 커밋이 생성된다



## - rebase

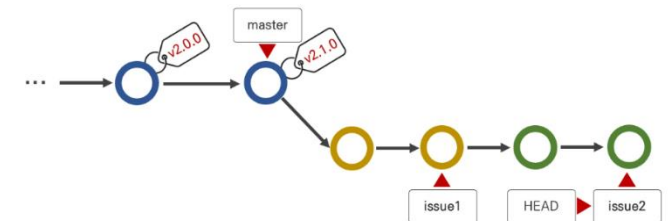
rebase를 통해서 브랜치의 **base** 조정하기

**issue2**의 base를 **issue1**로 재배치한다.



**\$ git rebase [branch]**

**\$ git rebase issue1**



충돌 수정 후 재배치하기 (commit 대신)

**\$ git rebase --continue**

rebase 자체를 취소하기

**\$ git rebase --abort**