

# git merge

- Fast-Forward Merge
  - 3-way Merge
- Merge Conflict

머지는 분기된 이력을 병합하는 방법으로 하나의 브랜치 변경 이력을 다른 브랜치에 병합한다.

다음은 병합의 일반적인 절차를 설명한다.

Step 1) 병합될 브랜치로 전환

- 본 예는 Feature → master 병합을 고려한다.

```
git checkout master
```

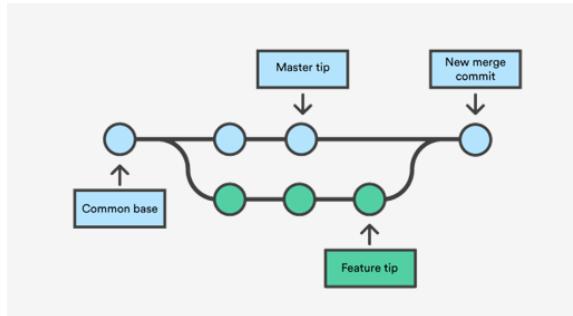
Step 2) origin 동기화

```
git pull origin
```

Step 3) 머지

머지 명령을 이용해 Feature 브랜치를 master 브랜치로 병합한다. --no-ff 옵션은 머지 커밋을 생성하고 싶은 경우 지정한다.

```
git merge [--no-ff] Feature
```



Step 4) Feature branch 삭제

```
git branch -d Feature
```

## Fast-Forward Merge

아래 그림과 같이 머지 대상 (master)에 변경 이력이 없는 상태에서 Feature의 변경 이력을 병합하는 경우를 의미한다.

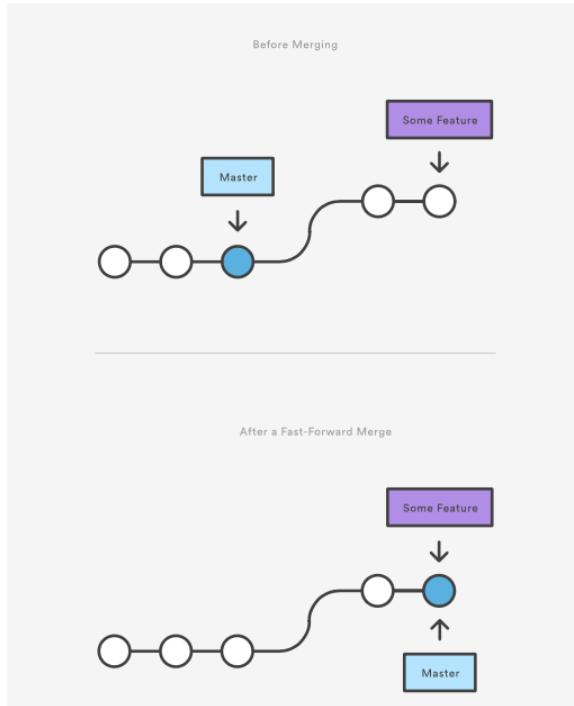
### Git 가이드

#### 1. Git 시작하기

- Git 저장소
  - git init
  - git clone
  - git config
- 변경 저장하기
  - git add
  - git commit
  - git diff
  - git stash
  - .gitignore
- 저장소 검색하기
  - git status
  - git log
  - git tag
  - git blame
- 변경 취소하기
  - git 실행 취소
  - git clean
  - git revert
  - git reset
  - git rm
- Rewriting history
  - git commit -- amend
  - git rebase
  - git reflog

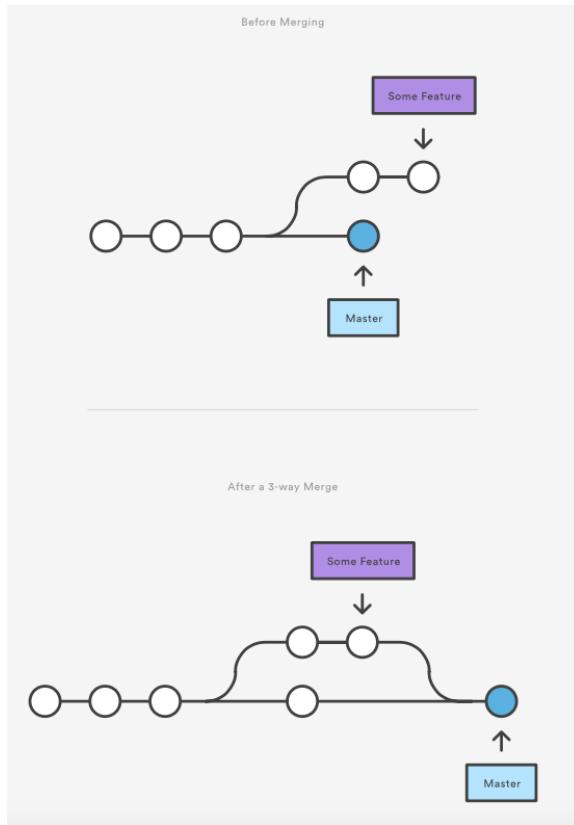
#### 2. Git 협업하기

- 동기화하기
  - git remote
  - git fetch
  - git pull
  - git push
- 브랜치 사용하기
  - git branch
  - git checkout
  - git merge
  - 병합 충돌 해결하기 (Merge conflicts)
  - 병합 전략 (Merge strategies)
- Pull request 만들기



### 3-way Merge

머지 대상 (본 예의 경우 master)에 변경 이력이 존재하는 상태에서 Feature의 변경 이력을 병합하는 경우를 의미한다.



### Merge Conflict

3-way merge의 경우 동일한 파일에 같은 지점이 변경된 경우 병합 대상 (본 예의 경우 Feature 브랜치)과 master를 병합할 때 충돌이 발생할 수 있다.

이 때 git은 고유한 방법으로 해당 파일 위치에 출동을 표시한다.

```
$ cat merge.txt
<<<<< HEAD
this is some content to mess with (    )
content to append
=====
totally different content to merge later
>>>>> new_branch_to_merge_later
```

일반적인 해소 절차는 관련 페이지를 참조한다.