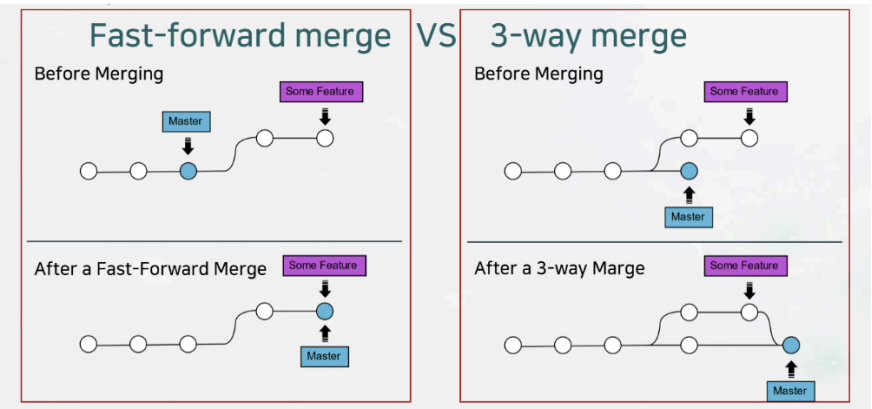
22~24

브랜치 병합

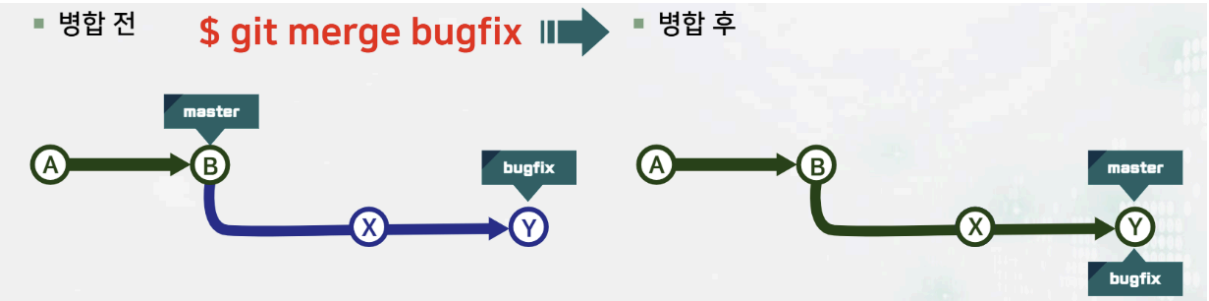
병합 : 두 개의 브랜치를 하나로 모으는 과정

- fast-forward(빨리 감기) 병합
- 3-way 병합



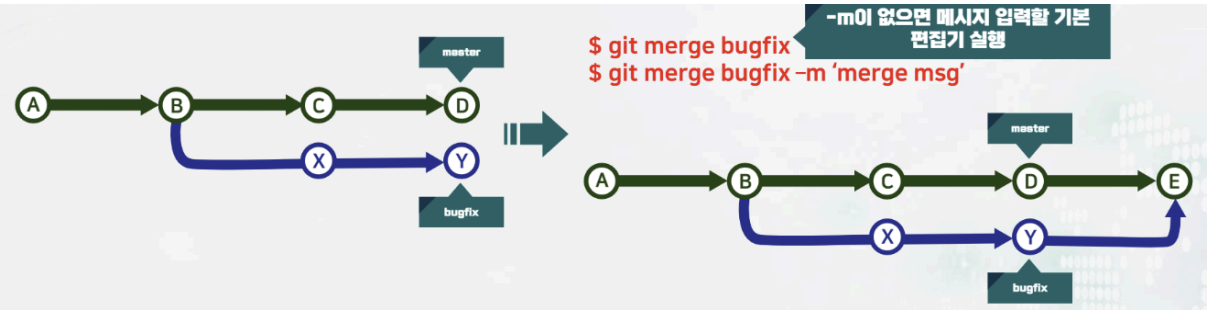
fast-forward 병합

- 조건 :
  - 현재 브랜치 master가 병합할 대상 커밋의 직접적인 뿌리(조상)가 되는 경우
  - bugfix 브랜치 이력이 master 브랜치 이력을 모두 포함하는 경우(일렬 상태)
- $ git merge 브랜치
- master 브랜치는 단순히 이동. 합칠 내용이 없음



3-way 병합

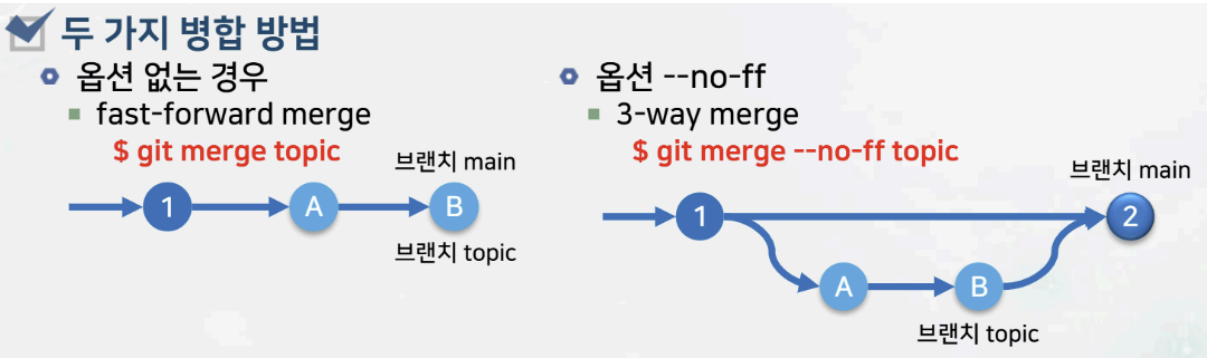
- 3-way 상태 : 두 분기가 갈라진 상태
  - 두 브랜치의 조상이 같은 경우, master와 bugfix를 하나로 통합할 필요
  - 새로운 커밋 'E' 생성 : 병합 완료 후 master로 통합된 이력이 생성
- $ git merge 브랜치
- $ git merge 브랜치 -m 'merge msg'



non fast forward 병합

일렬 상태에서 병합의 기본은 fast forward 병합

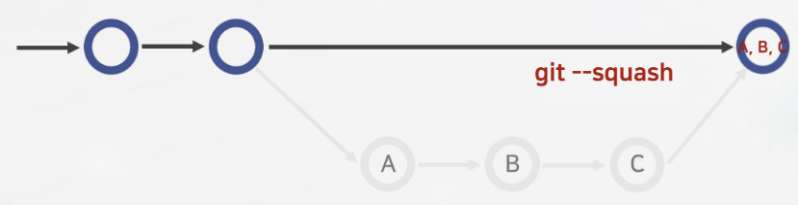
3-way 병합을 하고 싶을 때 : $ git merge - -no-ff 브랜치



병합과 옵션

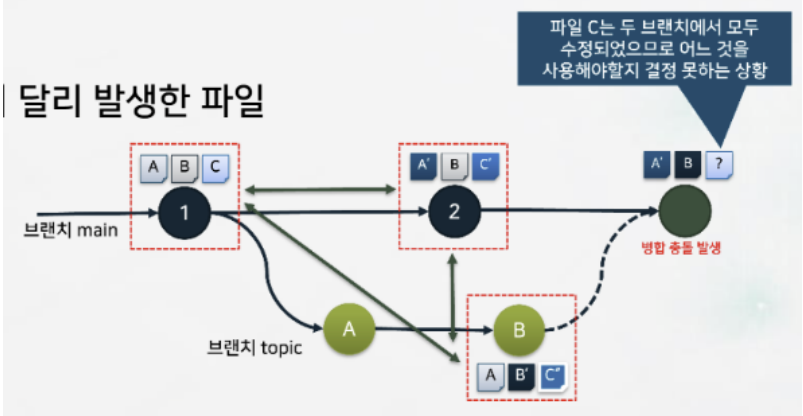
$ git merge 브랜치

- $ git merge --no-ff 브랜치 : 무조건 3-way 병합
- $ git merge --ff-only 브랜치 : 상태가 fast-forward인 일렬상태에서만 병합 진행
- $ git merge --squash 브랜치 : 현재 브랜치에 병합 대상과의 합치는 커밋을 하나 생성해 병합(병합되는 브랜치는 사용하지 않고 그대로 남음)
    - 커밋 이력, 병합되는 브랜치 이력 남기지 않음



병합 충돌

발생 이유 : 두 브랜치 모두에서 변경 사항이 달리 발생한 파일 때문



(이해를 위한 chatGPT 도움)

1️⃣A 파일 변화 — 충돌 아님

브랜치 topic에서만 A가 수정됨 → A'

main에서는 A가 바뀌지 않음.

👉 이런 경우 Git은 자동 병합 가능

→ 충돌 X

2️⃣ *B 파일 변화 — 충돌 아님*

브랜치 main에서만 B가 수정됨 → B′

topic에서는 B가 바뀌지 않음.

👉 이것도 자동 병합 가능

→ 충돌 X

3️⃣ ❗ *C 파일 변화 — 충돌 발생*

main에서 C → C' 로 수정됨

topic에서도 C → C'' 로 수정됨

*즉, **같은 파일 C를 두 브랜치가 각각 다르게 수정함***

→ 둘 중 무엇을 선택해야 할지 판단 불가

병합 충돌 해결

- 충돌한 파일 내부 표시 :

  <<<<<<< HEAD

  현재 브랜치 HEAD의 수정 내용

  ==========

  병합되는 브랜치 feat/list의 수정 내용

  >>>>>>> feat/list

- 병합 취소 후 다시 병합
  - 병합 취소 : $ git merge - -abort
  - 다시 병합 : $ git merge feat/list

- 직접 수정 후 저장
  - 충돌 표시 모두 제거
  - 추가, 커밋 다시