11주차 OSS 필기 정리

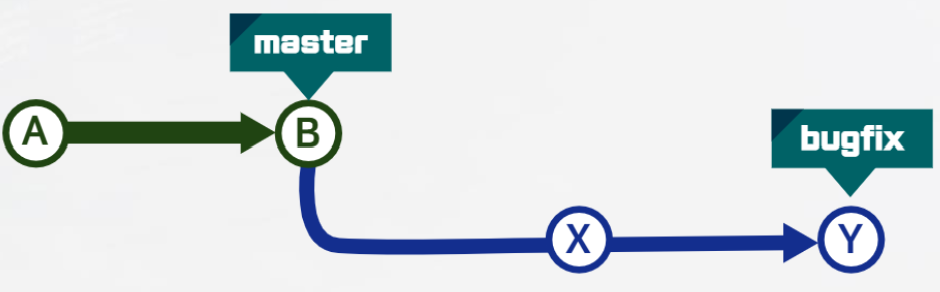
#22

<Main keyword> 병합 : 두 개의 브랜치를 하나로 모으는 과정

* Fast-forward 병합 vs 3-way 병합

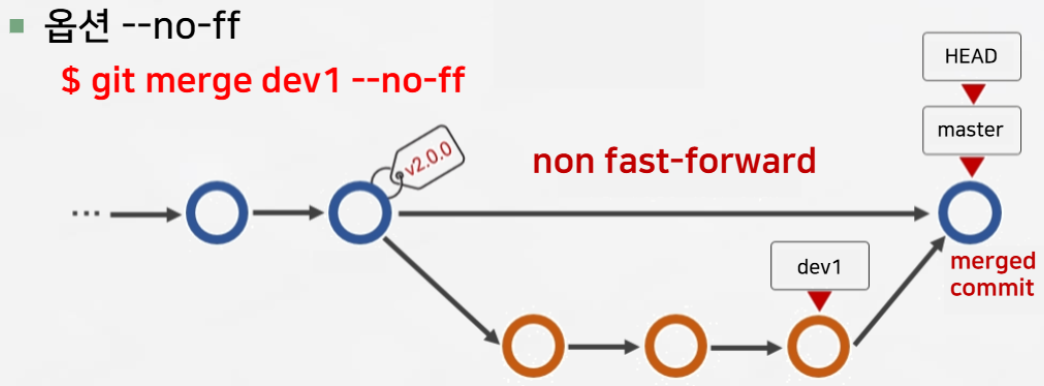
[ 두 브랜치의 일렬 상태에서 2가지 병합 ]

1. Fast-forward 병합 (병합 기본)



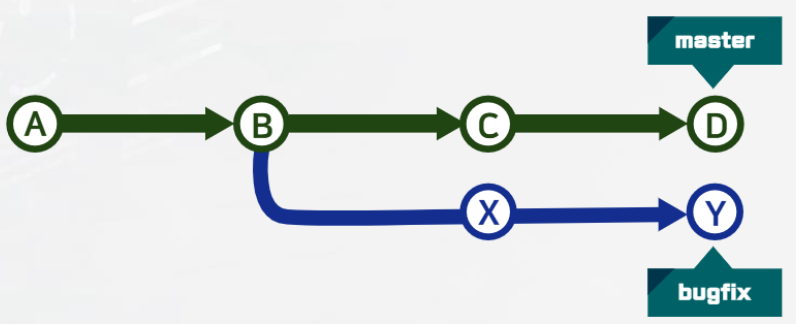
* 뒤에 있는 merge가 앞으로 간다고 봄, 간단히 두 브랜치가 일렬 상태
* 커밋 수 증가x, 단순히 브랜치가 이동
* Bugfix 브랜치 이력이 master 브랜치 이력을 모두 포함
* $ git merge bugfix : 브랜치 master에서 병합 명령
* $ git merge --f-only {병합할\_브랜치\_명} : 상태가 fast-forward인 일렬 상태에서만 병합 진행

1. Non fast forward 병합



* $ git merge dev1 –no-ff : 마스터에서 dev1을 non fast forward로 병합(3-way 병합)
* 병합된 브랜치가 그대로 남아서 실행한 작업 확인 및 브랜치 관리 면에서 더 유용
* $ git merge --squash hotfix : 커밋 이력과 병합되는 브랜치 이력도 남기지 않음, 새로운 커밋에 상채 브랜치의 내용을 모두 합해 새로운 커밋으로 병합

3-way 병합



* 두 갈래길에서 새로운 커밋 발생, 두 병합(merge)이 합쳐짐
* 병합 완료 후, 통합 브랜치인 ‘master’ 브랜치로 통합된 이력이 생성
* $ git merge bugfix (-m ‘merge msg’) : -m이 없으면 메시지 입력할 기본 편집기 실행

#24

<Main keyword> 병합 충돌

[ 충돌 기준 ]

* 파일 충돌 없음 -> 수정되지 않거나 한쪽 브랜치만 수정되면
* 파일 충돌 발생 -> 두 브랜치 모두에서 변경 사항이 달리 발생한 파일

[ 충돌 해결 ]

* 직접 파일 내용을 수정 후 추가, 커밋
* $ git merge –abort : 병합 취소
* $ git merge feat/list : 다시 병합

[ 충돌한 파일 내부 ]

<<<<<<< HEAD

현재 브랜치 HEAD의 수정 내용

==========

병합되는 브랜치 feat/list의 수정 내용

>>>>>>> feat/list