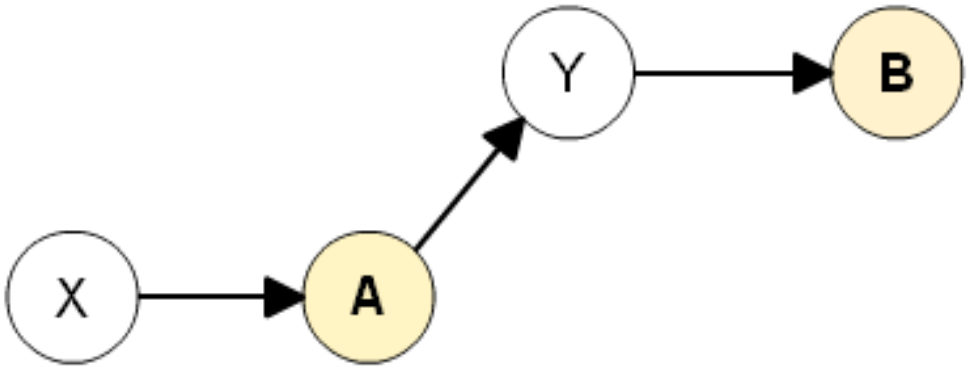
Bitbucket Merge strategies(병합 전략) 설정

- fast-forward 란 무엇인가?
  - fast-forward인 관계
  - fast-forward가 아닌 관계
- Merge strategies
  - 1.Merge commit (옵션 : --no-ff)
  - 2.Merge commit (옵션 : --ff)
  - 3.Merge commit (옵션 : --ff-only)
  - 4.Merge commit (옵션 : --squash)
  - 5.Merge commit (옵션 : --squash --ff-only)
- Rebase 란 무엇인가?
  - 6.Merge commit (옵션 : rebase + merge --no-ff)
  - 7.Merge commit (옵션 : rebase + merge --ff-only)

fast-forward 란 무엇인가?

2개의 커밋 A와 B가 있을때에 커밋A에 따른 히스토리가 커밋B에 따른 히스토리에 전부 포함되어 있는 경우
2개의 커밋은 fast-forward관계이다 or 커밋A는 커밋B에 fast-forward한다라고 말한다.

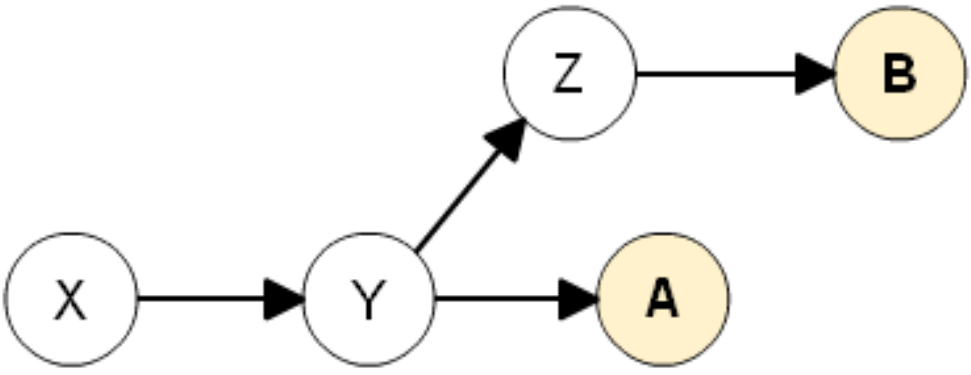
fast-forward인 관계



- 커밋 A의 히스토리 : X > A
- 커밋 B의 히스토리 : X > A > Y > B

커밋B의 히스토리에 커밋A의 히스토리가 모두 포함되어 있다.

fast-forward가 아닌 관계



- 커밋 A의 히스토리 : X > Y > A
- 커밋 B의 히스토리 : X > Y > Z > B

커밋B의 히스토리에 커밋A의 히스토리가 포함되어 있지 않다.（A의 커밋이 포함되어 있지 않다.）

2014 - 2023 CURVC Corp. All rights reserved

*참고페이지 : http://blog.naver.com/PostView.nhn?blogId=parkjy76&logNo=220308638231&categoryNo=73&parentCategoryNo=0&viewDate=&currentPage=1&postListTopCurrentPage=1&from=postView

Merge strategies

프로젝트,저장소 에서 Merge 정책을 설정

설정 > Merge strategies > ON/OFF, Default Merge 정책 설정 가능

옵션1. --no-ff (non-fast-forward)

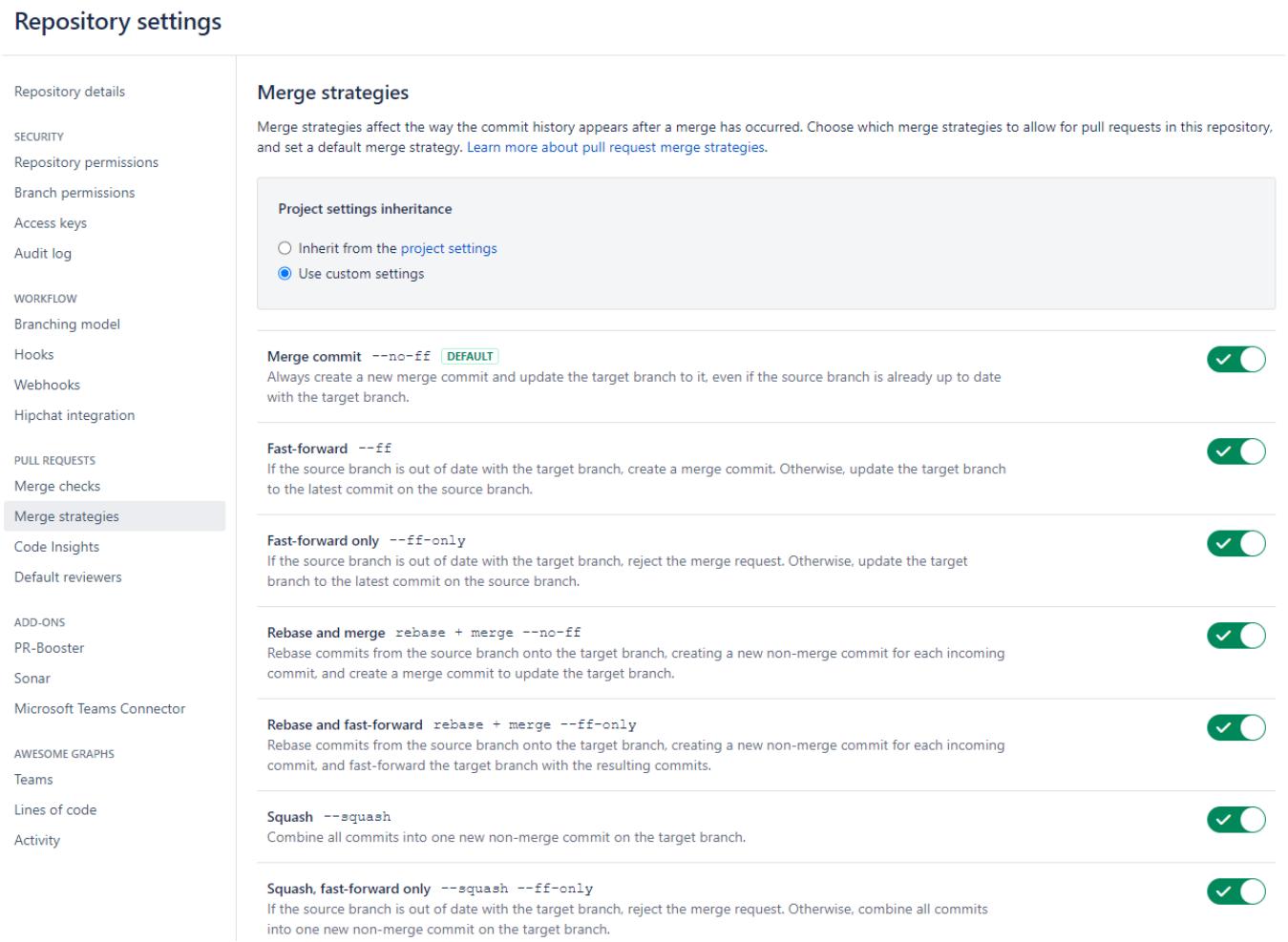
머지 작업 시 필히 머지 커밋을 생성하여 브랜치가 있었다는 정보가 남는다.

옵션2. --ff (fast-forward)

로그가 하나로 되어 브랜치가 있었다는 정보가 없어진다.

브랜치의 존재를 로그에 남길 필요가 없어 그대로 커밋해도 문제가 없는 경우에 사용.

옵션3. --squash
브랜치가 있었다는 정보 커밋 로그가 모두 남지 않는다.
브랜치에서 행한 작업이 작아 1커밋으로 정리해버려도 문제가 없는 경우 사용.

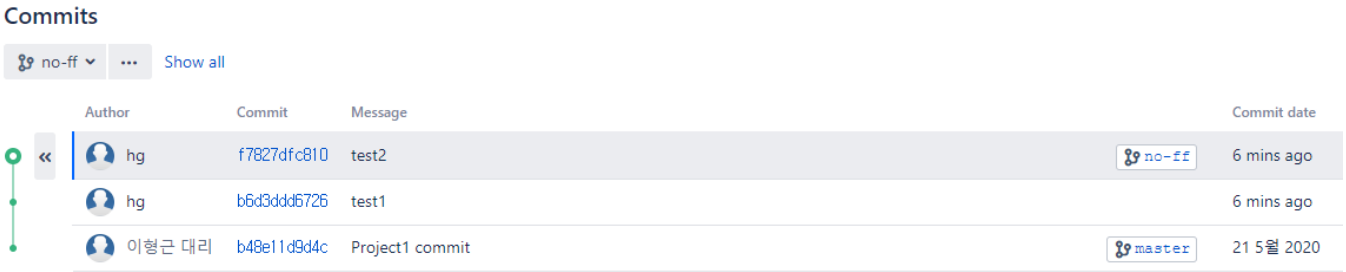


2014 - 2023 CURVC Corp. All rights reserved

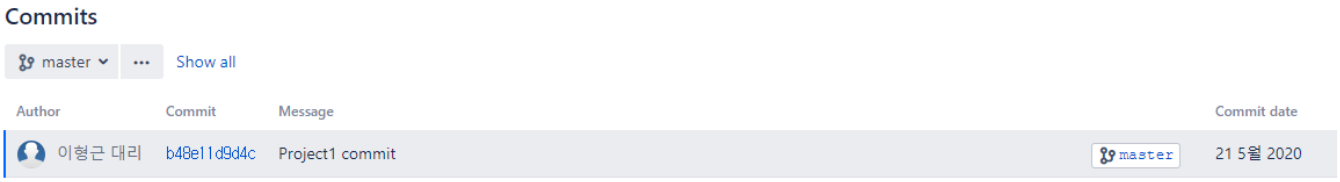
1.Merge commit (옵션 : --no-ff)

- fast-forward의 관계라도 필히 머지 커밋을 생성
- non fast-forward 병합 을 실행하면, 브랜치가 그대로 남기 때문에 그 브랜치로 실행한 작업 확인 및 브랜치 관리 면에서 좋음 (변경 이력 모두 확인 가능)

1) no-ff 브랜치 Commit 상태



2) master 브랜치 Commit 상태

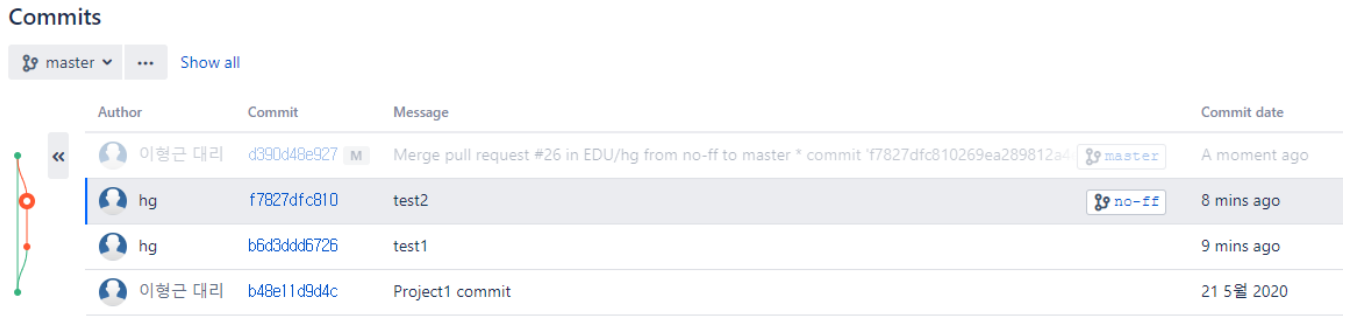


3) --no-ff 옵션으로 Merge 진행

- master 브랜치에 Merge된 no-ff 브랜치 작업 히스토리 확인 가능
- 필히 머지 커밋 생성



Master 브랜치 Commit

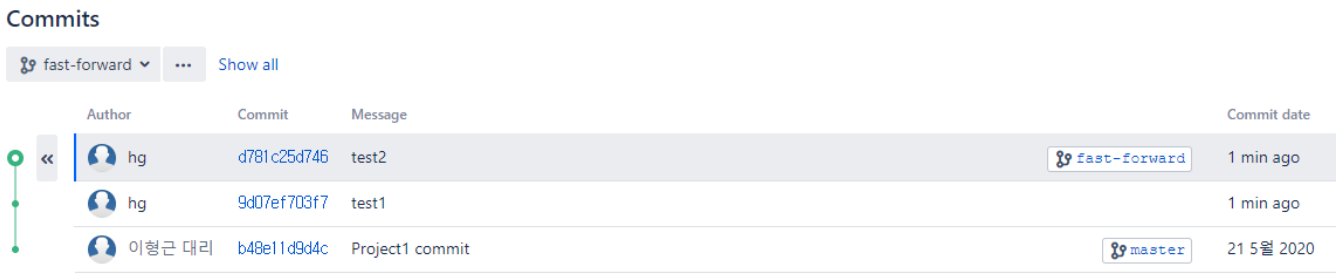


2.Merge commit (옵션 : --ff)

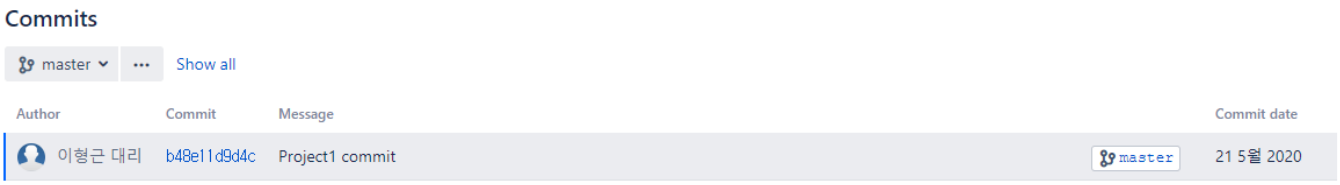
- 처음 설명 확인 필요
- fast-forward의 관계이면 머지 커밋이 생성 되지 않음
- fast-forward의 관계가 아닐 때 머지 커밋을 생성

2014 - 2023 CURVC Corp. All rights reserved

1) fast-forward 브랜치 Commit 상태

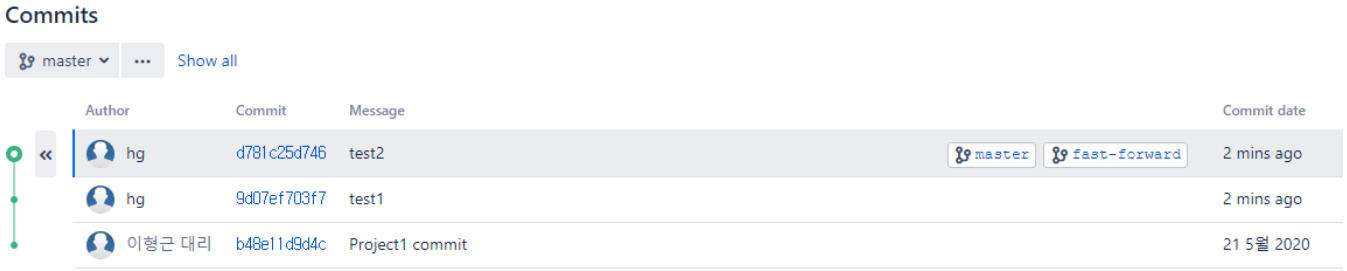


2) master 브랜치 Commit 상태

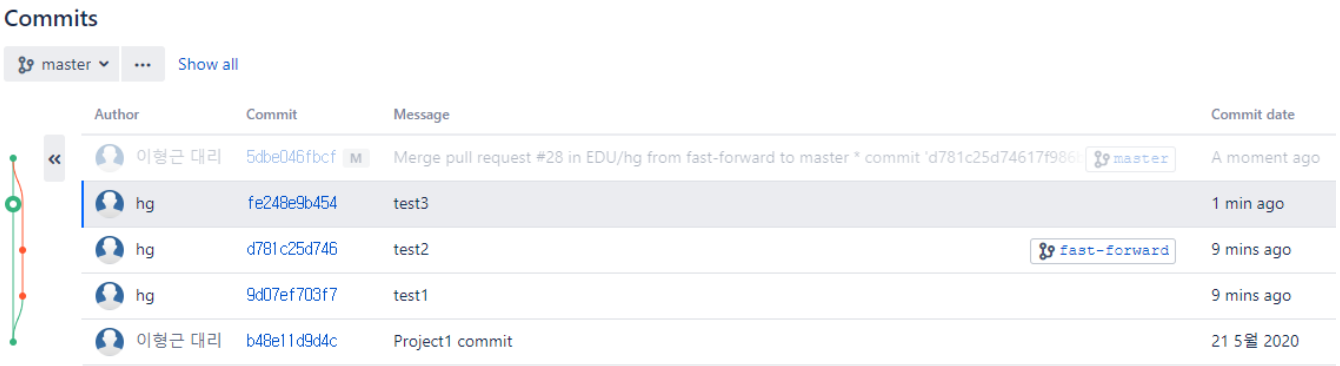


3) --ff 옵션으로 Merge 진행

- **fast-forward의 관계 > 머지 커밋이 생성 되지 않음



- **fast-forward의 관계가 아닐 때 > 머지 커밋 생성



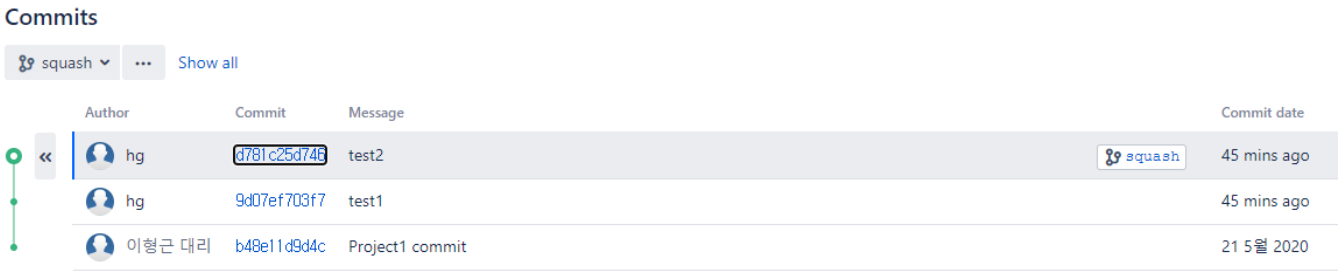
3.Merge commit (옵션 : --ff-only)

2014 - 2023 CURVC Corp. All rights reserved

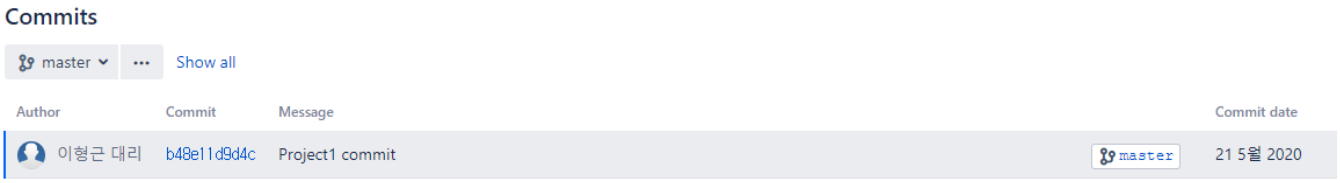
- fast-forward의 관계일 경우에만 머지 진행 (머지 커밋 생성되지 않음)
- fast-forward 관계가 아니면 머지가 진행 되지 않음

4.Merge commit (옵션 : --squash)

1) squash 브랜치 Commit 상태

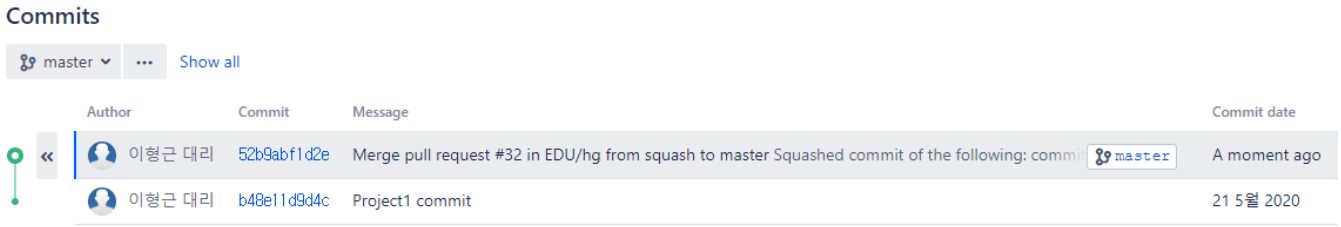


2) master 브랜치 Commit 상태



3)Master 브랜치 Commit

- 머지 시 squash브랜치 commit(test1,test2) 는 머지 커밋 1개로 합쳐서 master 머지 됨



5.Merge commit (옵션 : --squash --ff-only)

- fast-forward의 관계일 경우에만 머지 진행
- fast-forward 관계가 아니면 머지가 진행 되지 않음

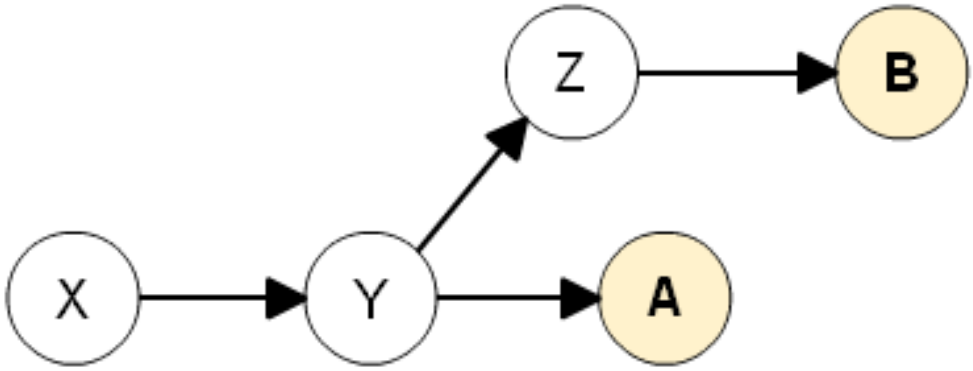
Rebase 란 무엇인가?

두 브랜치를 합치는 두 번째 방법인 Rebase는 Merge와는 다르게 이름 그대로 브랜치의 공통 조상이 되는 base를 다른 브랜치의 **커밋 지점**으로 바꾸는 것입니다.

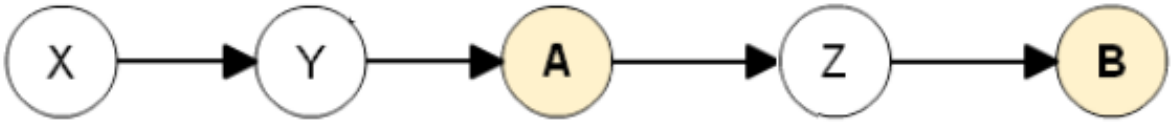
Master 현재 위치가 base가 되고 뒤에 Feature브랜치 Commit이 저장

- 커밋 A : **Master** / 커밋 B : **Feature브랜치** 일 경우

2014 - 2023 CURVC Corp. All rights reserved

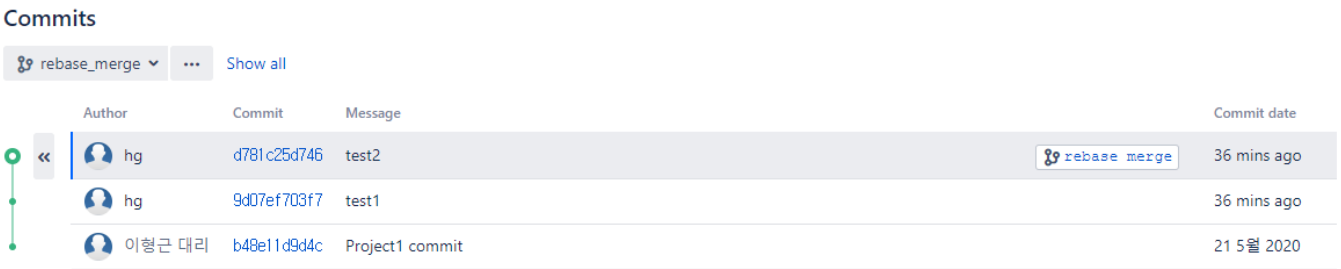


- Rebase Merge 시 master 브랜치

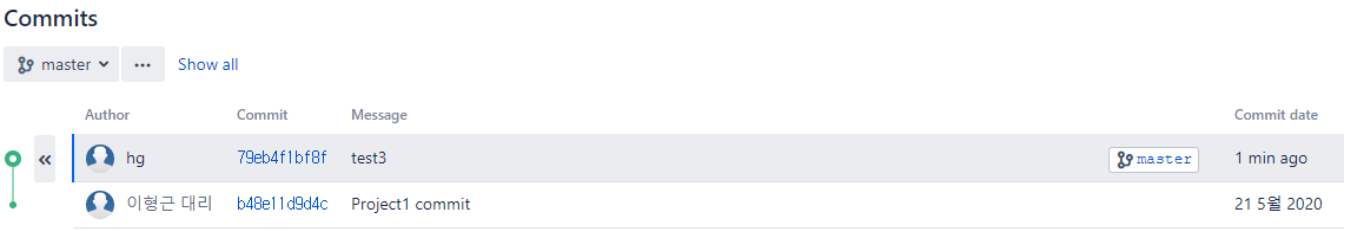


6.Merge commit (옵션 : rebase + merge --no-ff)

1) rebase_merge 브랜치 Commit 상태



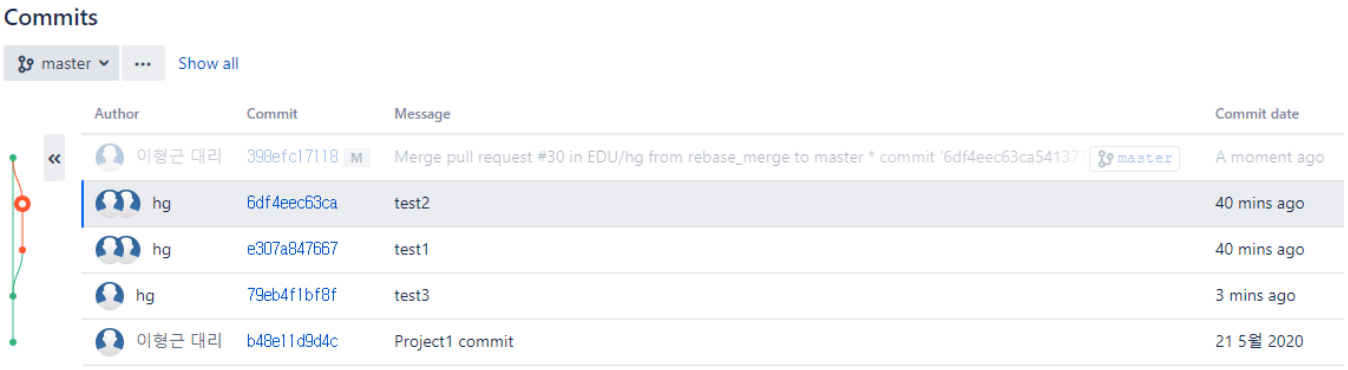
2) master 브랜치 Commit 상태



3) rebase + merge --no-ff 옵션으로 Merge 진행

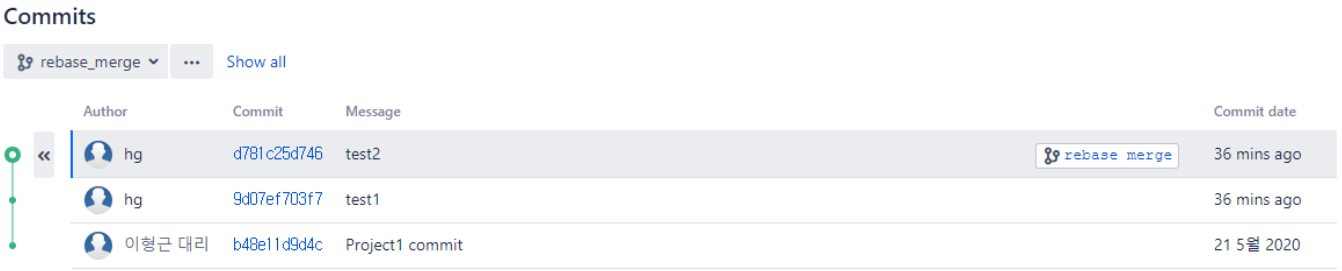
- master 브랜치 base : test3 + rebase_merge브랜치 내용 합쳐짐 (-no-ff 옵션으로 M 머지커밋 생성)

2014 - 2023 CURVC Corp. All rights reserved

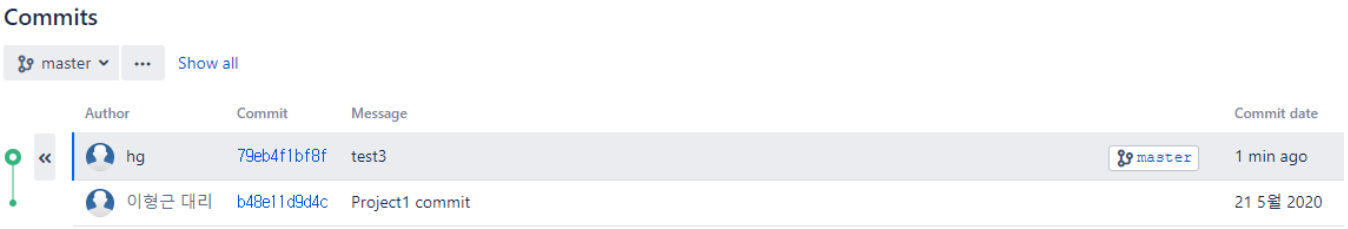


7.Merge commit (옵션 : rebase + merge --ff-only)

1) rebase_merge 브랜치 Commit 상태

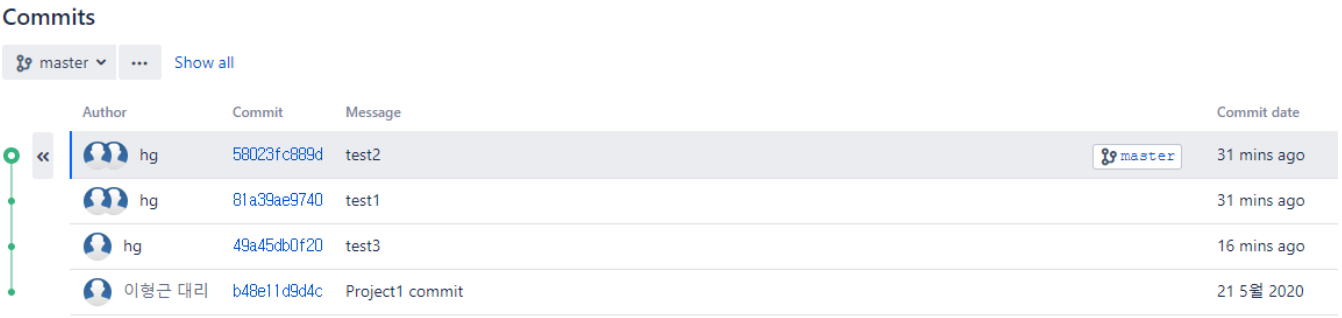


2) master 브랜치 Commit 상태



3) rebase + merge --ff-only 옵션으로 Merge 진행

- master 브랜치 base : test3 + rebase_merge브랜치 내용 합쳐짐



2014 - 2023 CURVC Corp. All rights reserved

2014 - 2023 CURVC Corp. All rights reserved