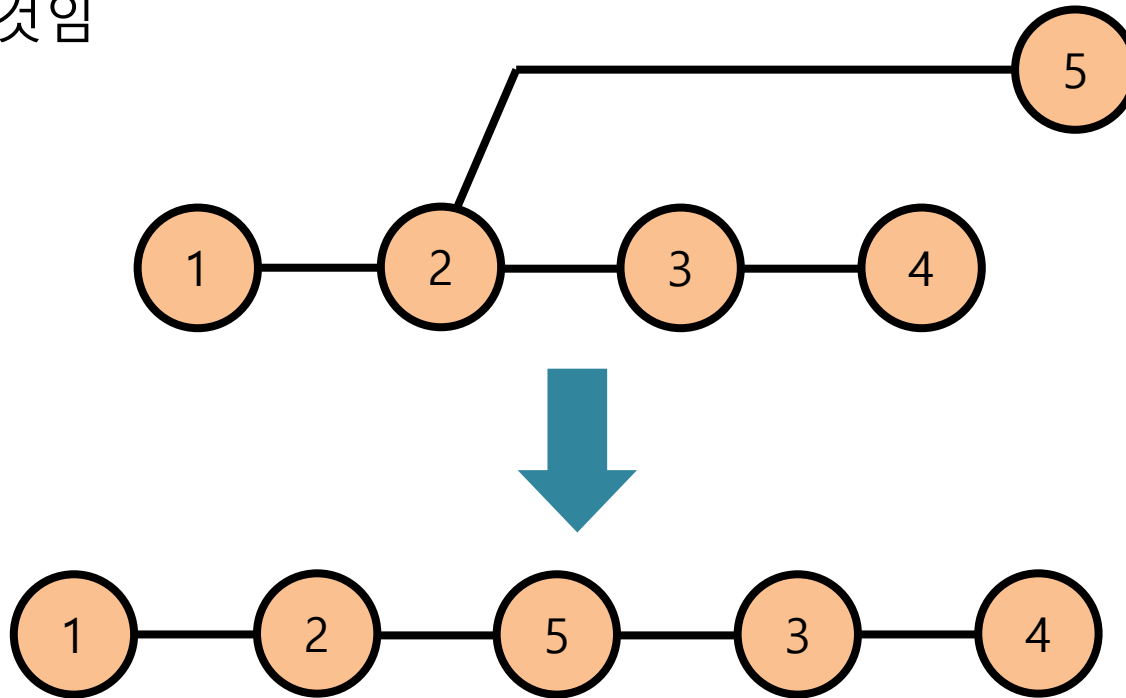


GUI 환경에서의 rebase, amend

세종대학교 이은상

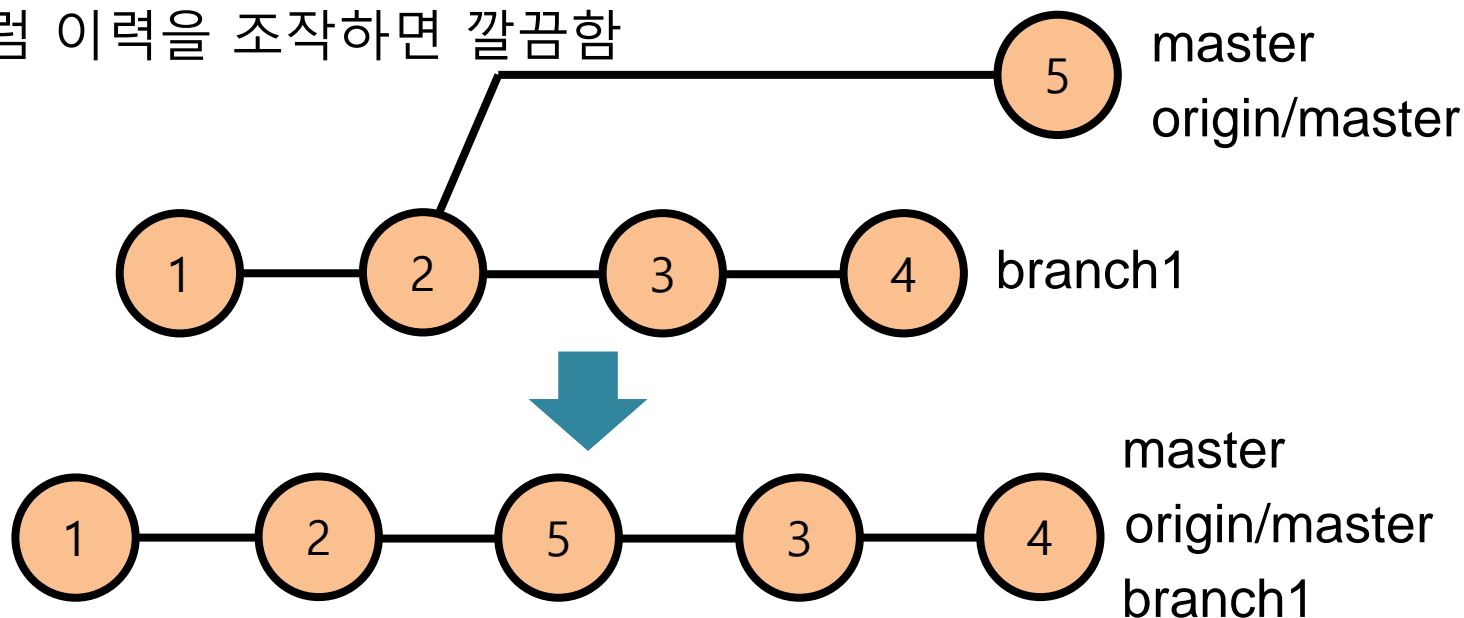
Rebase

- rebase는 예전 커밋을 새 커밋으로 이력을 조작하는 것임
- 아래 그림에서 커밋3, 4는 원래 base가 2임. 이를 마치 base가 5인 것처럼 base를 바꾸는(rebase)것임
- 즉, 내가 작업한 커밋3, 4의 base를 가장 최신 커밋(커밋5)로 바꾸어 주는 것임



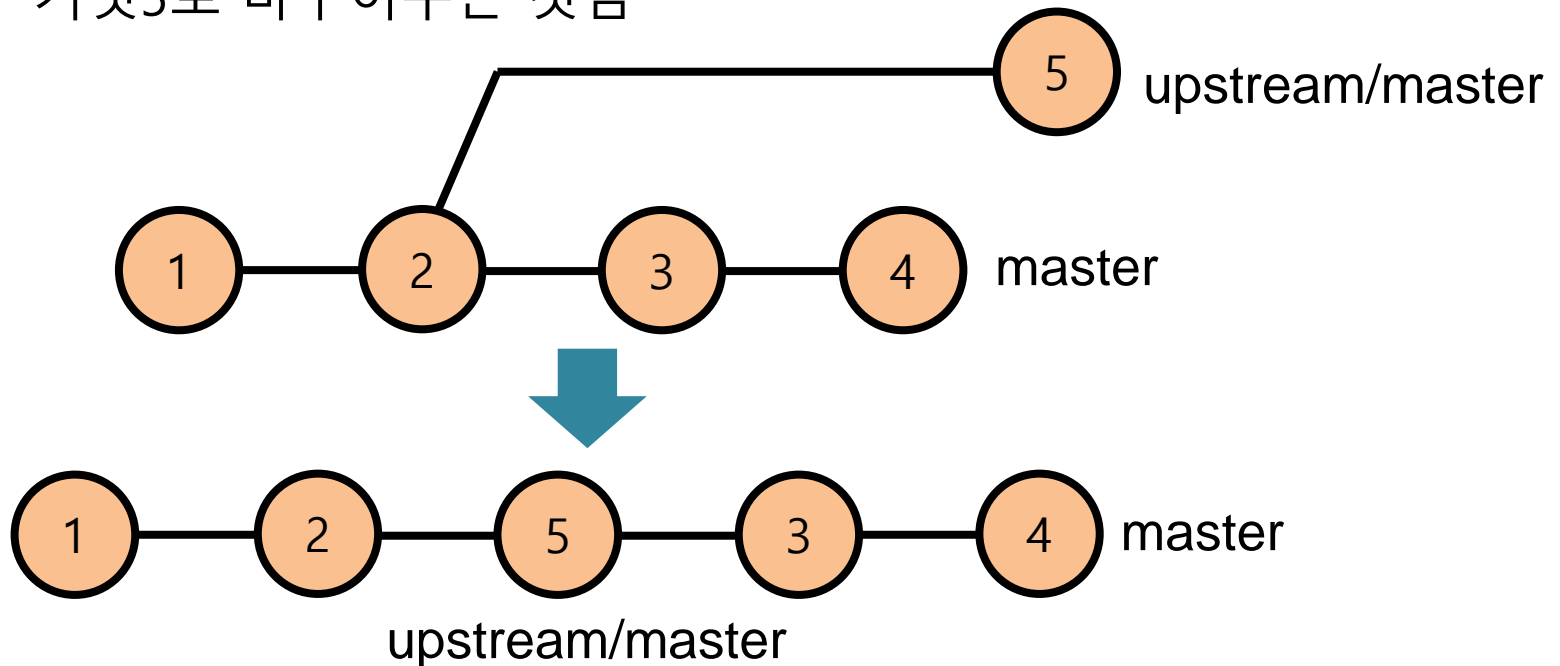
Rebase

- Rebase는 크게 두 가지 상황으로 나뉠 수 있음
- Case1) 한 원격저장소 내의 서로 다른 브랜치에서 작업하는 경우
 - 아래 그림처럼 커밋2를 기준으로 branch1에서 2개의 커밋을 한 후 master에 합치려고 보니 master 브랜치에 최근 한 커밋이 추가된 것을 확인함
 - 이 경우 마치 최근 커밋5를 기준으로 거기서 추가로 커밋한 것처럼 이력을 조작하면 깔끔함



Rebase

- Case2) Fork를 하여 작업하는 상황
 - 아래 그림처럼 fork를 하여 master 브랜치에서 커밋 2개를 한 후 pull request를 하려고 보니 원본저장소 master 브랜치에 최근 한 커밋이 된 것을 확인함
 - 마치 최근 커밋5를 base로 거기부터 작업하여 커밋 2개(커밋3, 4)를 한 것처럼 이력을 조작하면 깔끔함. 즉, 커밋3, 4의 base를 커밋5로 바꾸어주는 것임

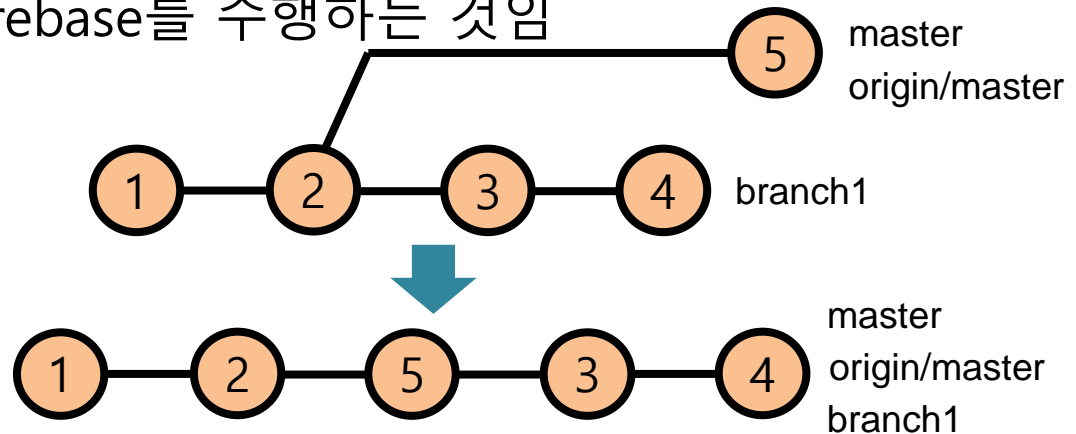


Upstream

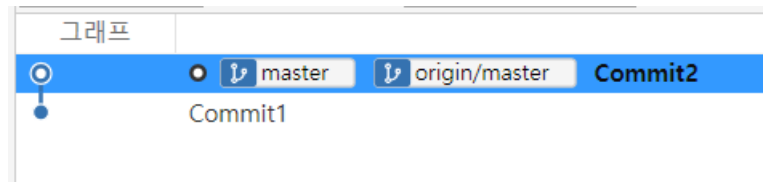
- 외부계정에서 원본저장소를 fork하면서 새로운 원격저장소를 만든 상황임
- Fork한 시점까지의 모든 히스토리는 외부계정에서 알 수 있지만 그 다음에 원본저장소에 무슨 일이 일어났는지 모름. 원본저장소와 fork한 원격저장소는 이미 주소까지 바뀌어 서로 다른 원격저장소이기 때문
- 따라서, 내 원격저장소에서 다른 원본저장소의 히스토리도 보는 것이 필요함
- 외부계정이 원본저장소의 커밋 히스토리를 받을 때 원본저장소에 대해 "upstream"이라는 닉네임을 사용함

Rebase 실습 (서로 다른 브랜치)

- 실습의 목표는 다음 rebase를 수행하는 것임

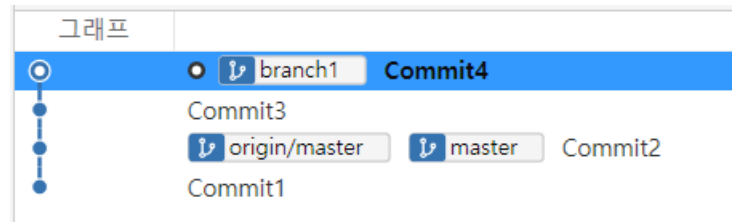


- 먼저, 원격저장소와 연결된 빈 로컬저장소를 생성함
 - 폴더명: GitDirectory1
 - 원격저장소명: GitRepository1
- master 브랜치에서 새 파일을 만들어 커밋 2개를 하고 push함
 - 새 파일명: GitFile1.txt
 - 커밋1: Commit1, 파일 내용 Contents1 작성
 - 커밋2: Commit2, 파일 내용을 Contents2으로 수정

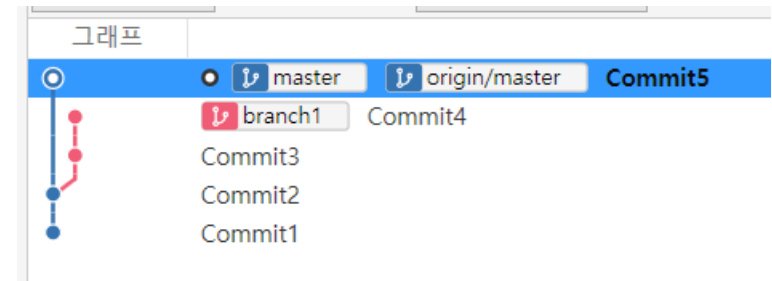
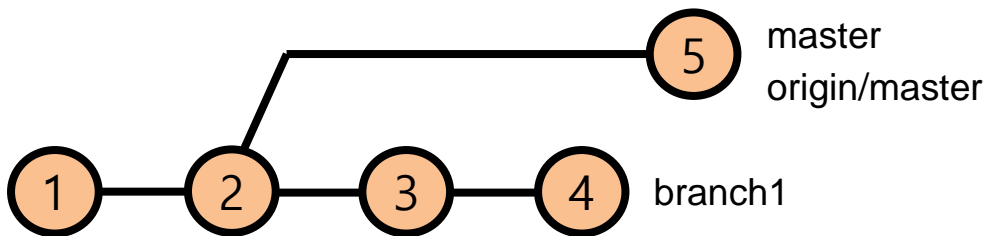


Rebase 실습 (서로 다른 브랜치)

- 그 후 branch1 브랜치를 만들어 거기서 커밋 2개를 추가로 함. 아직 push를 하지는 않음
 - 커밋3: 커밋명은 Commit3. GitFile1.txt의 내용을 Contents3로 수정
 - 커밋4: 커밋명은 Commit4. GitFile1.txt의 내용을 Contents4로 수정

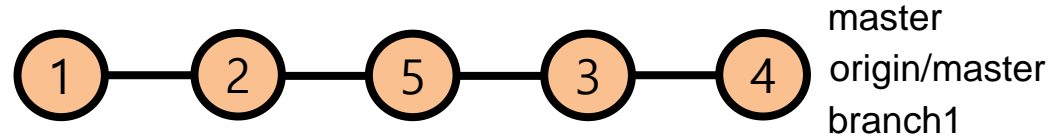


- 이제 master 브랜치로 체크아웃해서 커밋을 하나 작성함. 그 후 push를 하면 아래와 같은 상황이 생김
 - 커밋5: 커밋명은 Commit5. GitFile2.txt를 생성함. 내용은 Contents1.

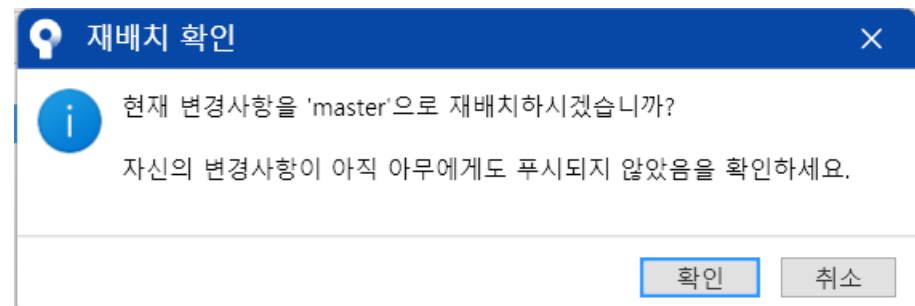
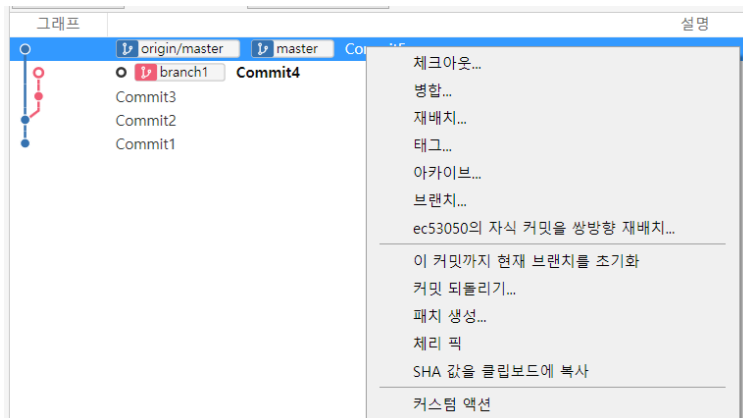


Rebase 실습 (서로 다른 브랜치)

- branch1의 내용을 master에 반영하려면 병합을 해야하는데 병합 시 커밋 이력이 예쁘지 않음
- 따라서, 아래와 같이 rebase를 한 후 push를 하고자 함

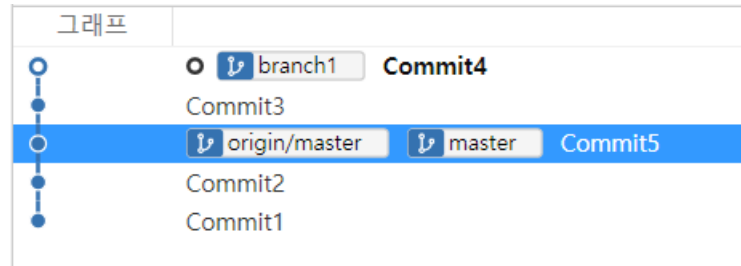


- 일단 branch1으로 체크아웃함. 새로운 베이스로 삼고 싶은 커밋 (master)에서 마우스 오른쪽 버튼 눌러 [재배치]를 클릭함. 재배치는 rebase의 한글 번역임
- 경고창이 나오면 확인을 클릭함

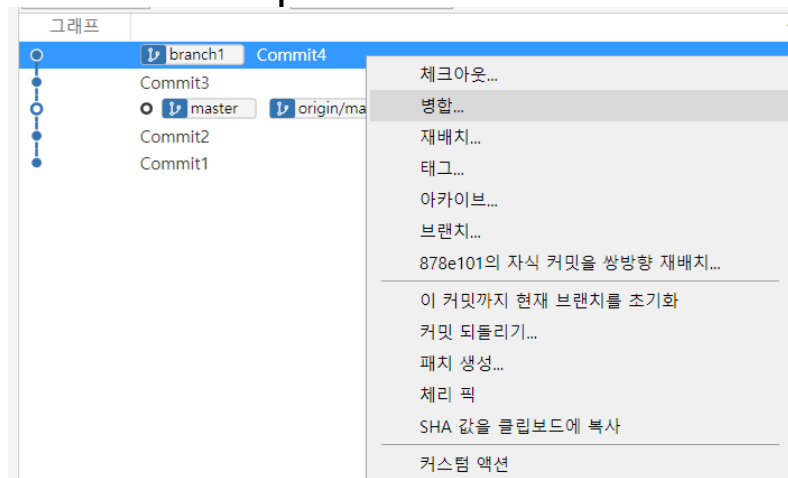


Rebase 실습 (서로 다른 브랜치)

- 그러면 그래프가 깔끔해진 것을 볼 수 있음

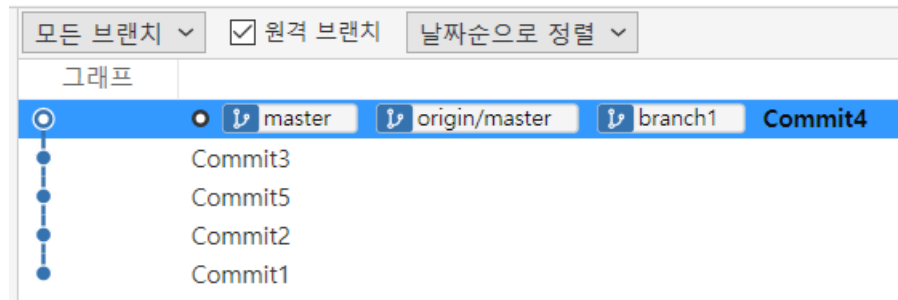
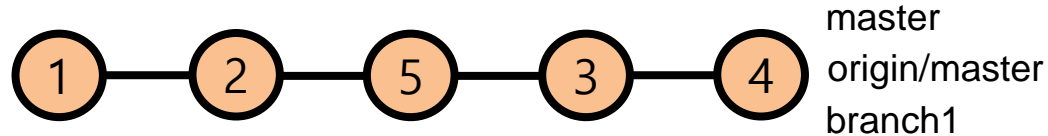


- 이제, master 브랜치를 기준으로 branch1을 master 브랜치에 병합하고자 함.
- 먼저, master 브랜치로 체크아웃함. 그 후 branch1에 오른쪽 마우스 클릭하고 병합 클릭함. 그 후 push까지 진행함



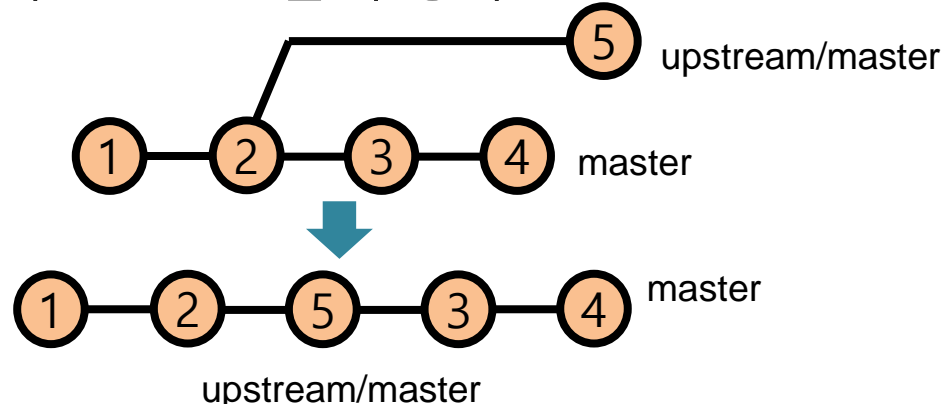
Rebase 실습 (서로 다른 브랜치)

- 그러면 커밋 이력이 깔끔하게 됨



Rebase 실습 (Fork)

- 실습의 목표는 다음 rebase를 수행하는 것임

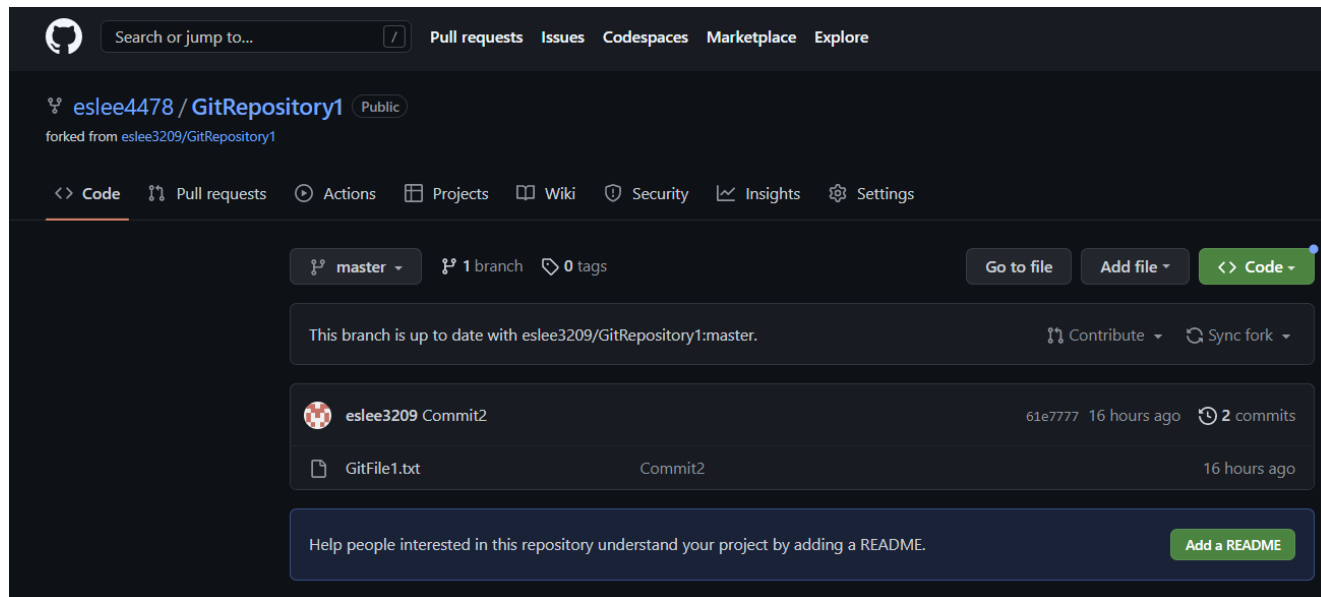


- 먼저, 주 계정으로 원격저장소와 연결된 로컬저장소를 새로 만들고 다음과 같이 커밋 2개를 생성함
 - 계정: eslee3209
 - 원격저장소 이름: GitRepository1
 - 로컬폴더 이름: GitDirectory1
 - GitFile1.txt을 생성하고 각 내용을 Contents1, Contents2로 하여 순서대로 커밋함
 - 커밋메시지: Commit1, Commit2



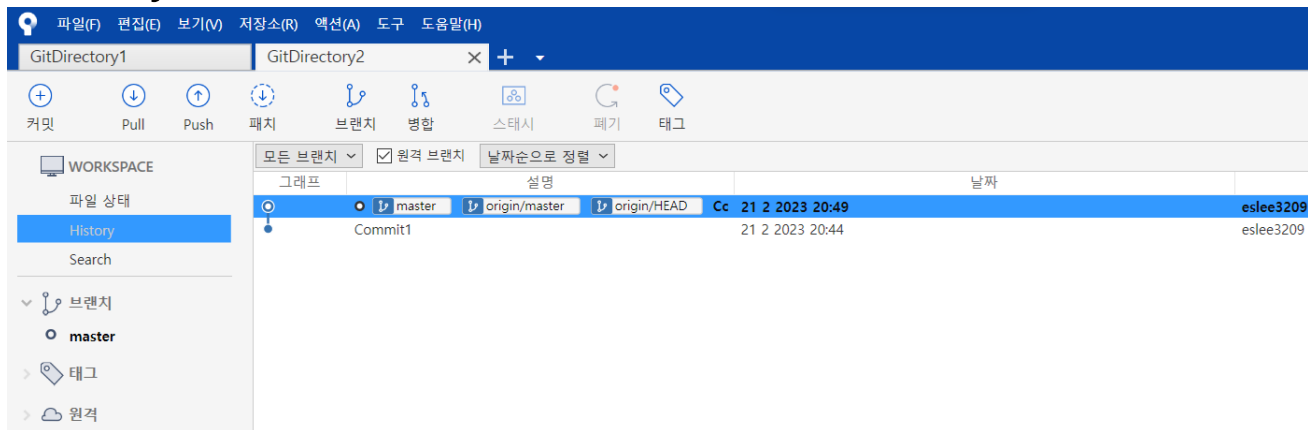
Rebase 실습 (Fork)

- 이제 부계정(eslee4478)으로 GitHub에 로그인하여 원본저장소를 fork함
- fork한 원격저장소 이름은 동일하게 GitRepository1로 함

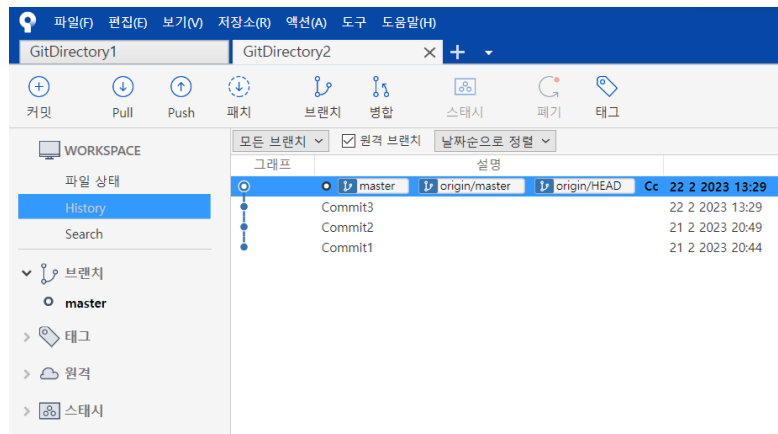


Rebase 실습 (Fork)

- 소스트리에서 fork한 원격저장소를 로컬에서 보기 위해 clone함. 이 때, 로컬저장소 폴더명은 GitDirectory2로 함(주계정이 사용하는 GitDirectory1 폴더와 구분 위해)



- 이제, 부계정으로 master 브랜치에 커밋 2개를 추가한 후 push함



Rebase 실습 (Fork)

- 소스트리에서 다시 주계정으로 로그인함. GitDirectory1에서 커밋 하나를 추가하여 커밋한 후 push까지 함
 - GitFile2.txt 파일 생성. 내용은 Contents1
 - 커밋 메시지: Commit5
- 그 후 다시 소스트리에서 부계정으로 로그인함. 원본저장소에 변경사항은 없는지 확인 위해 원본저장소를 로컬로 불러옴
 - [저장소]>[원격 저장소 추가]를 클릭함
 - 좌측에 [추가] 버튼을 클릭함
 - 원격 이름에는 'upstream'이라고 넣음. 이것은 원본저장소를 지칭하는 관용적 닉네임임
 - URL/경로에는 원본저장소 주소를 복사해서 적어줌. 그 후 확인을 눌러줌
 - 예) 원본저장소 주소: <https://github.com/eslee3209/GitRepository1.git>

Rebase 실습 (Fork)

- 그러면 upstream 원본저장소가 추가됨
- [확인]을 눌러 저장함

원격 저장소 정보

필요한 정보

원격 이름: upstream

☐ 디폴트 원격

URL / 경로: https://github.com/eslee3209/GitRepository1.git

추가 확장 통합

Remote Account:

Generic Account

Generic Host

▼

Legacy Account Settings:

호스트 종류: GitHub ▼

호스트 루트 URL: https://www.github.com

사용자명: eslee3209

확장 통합을 사용하면 Bitbucket 과 같은 외부 호스팅과 연계할 수 있습니다. 예를들어, 사이트에서 링크를 열 때 기존의 클론을 찾아 표시하거나, 그대로 pull 요청을 만들 수 있습니다.

확인

취소

저장소 설정

원격

고급

원격 저장소 경로

이름	경로
origin	https://github.com/eslee4478/GitRepository1.git
upstream	https://github.com/eslee3209/GitRepository1.git

추가

편집

제거

설정 파일 편집...

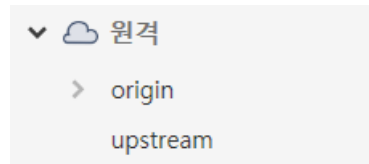
확인

취소

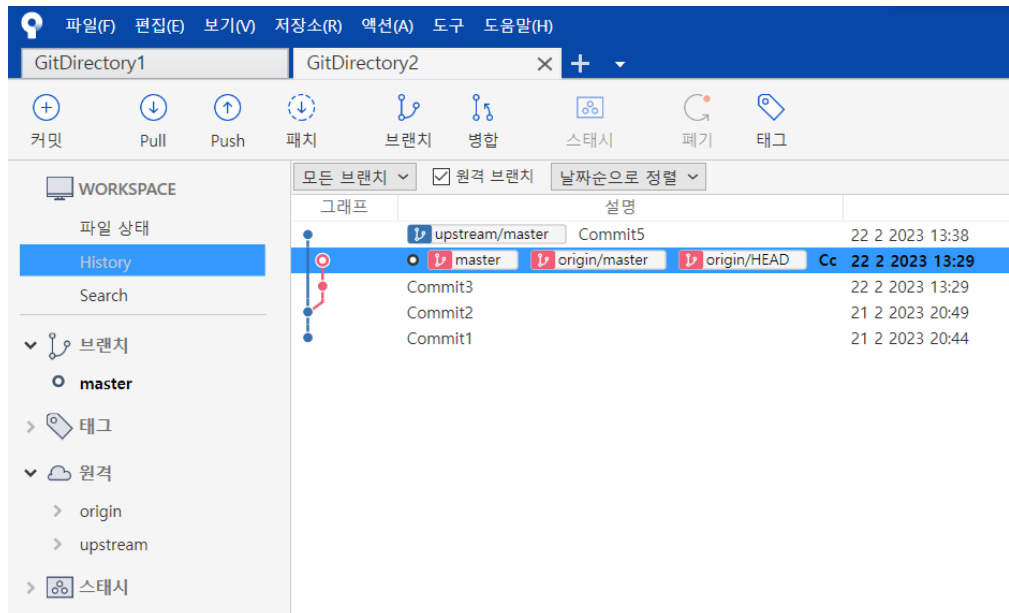
15

Rebase 실습 (Fork)

- 그러면 [원격] 세션에 upstream이 추가됨

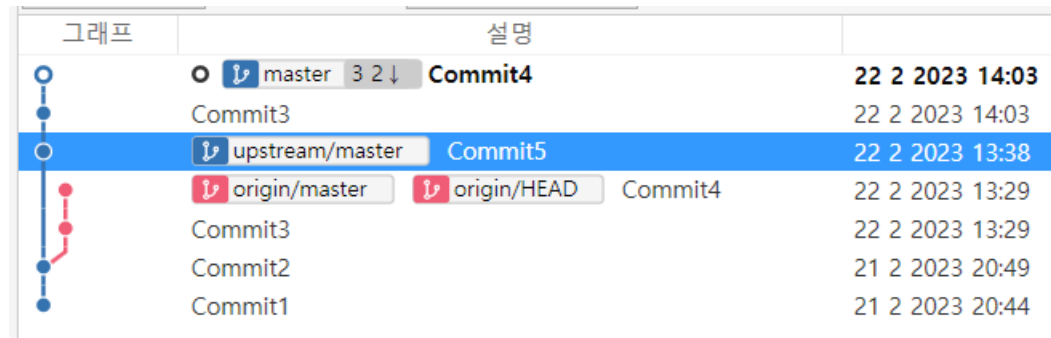


- 이제, upstream에서 마우스 우측 클릭하고 [upstream]에서 가져오기] 메뉴를 선택함. [upstream] 원본 저장소에 있는 커밋 히스토리를 가져오는 것임. 이것을 패치라고 함. 그래프가 업데이트 됨



Rebase 실습 (Fork)

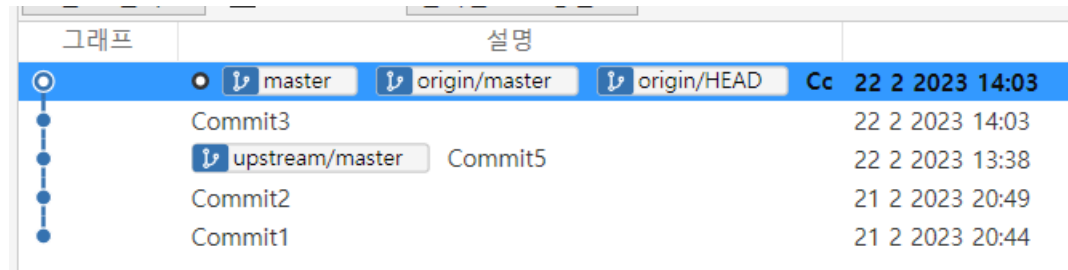
- 이 상태에서 병합을 하면 병합 커밋이 생겨서 예쁘지 않음. 따라서 rebase를 수행함
- 먼저, master 브랜치로 체크아웃함. upstream/master에서 마우스 오른쪽 클릭한 후 [재배치]를 클릭함



- 위와 같이 upstream/master를 기준으로 rebase가 성공적으로 된 것을 알 수 있음

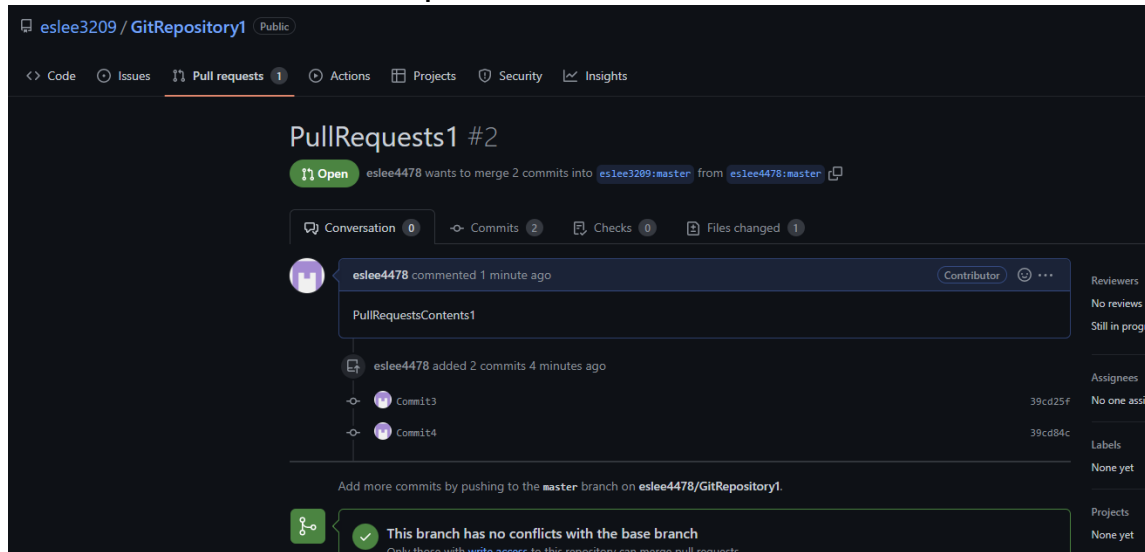
Rebase 실습 (Fork)

- 아직 origin/master에 반영이 되지 않았으므로 push를 하되 강제 푸시를 해줌
- 그러면 다음과 같이 그래프가 깔끔해짐



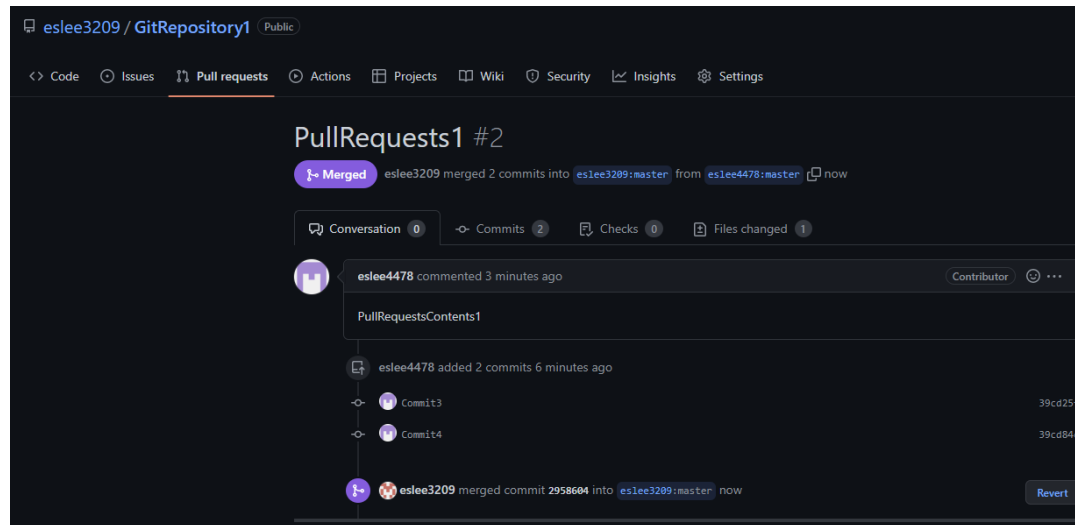
Rebase 실습 (Fork)

- 이제, pull request만 하면 됨
- 부계정으로 GitHub fork한 원격저장소(eslee4478/GitRepository1)로 들어가 pull request를 누름
 - pull request 이름: PullRequests1
 - pull request 상세내용: PullRequestsContents1

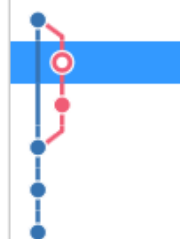






Rebase 실습 (Fork)

- 주계정으로 다시 GitHub 로그인하여 원본저장소에 들어가 승인을 함



- 소스트리에서 부계정으로 로그인하여 패치를 하여 그래프 업데이트 하면 다음과 같이 됨

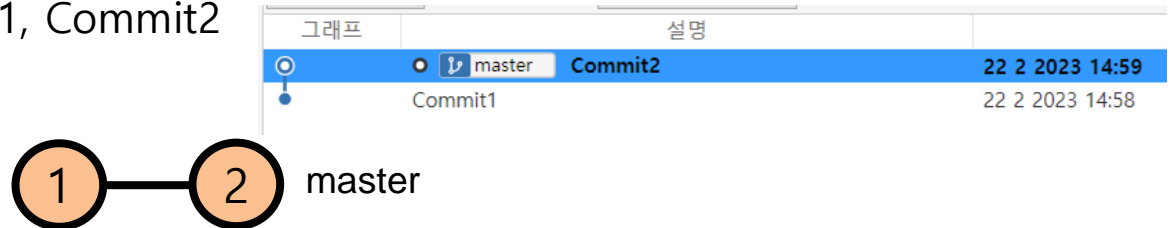
그래프	설명	
	 upstream/master Merge pull request #2 from eslee4478/master	22 2 2023 14:10
	 master  origin/master  origin/HEAD Commit4	22 2 2023 14:03
	Commit3	22 2 2023 14:03
	Commit5	22 2 2023 13:38
	Commit2	21 2 2023 20:49
	Commit1	21 2 2023 20:44

Amend

- 커밋을 하나 만들었는데 추가할 파일이 있다는 것을 뒤늦게 알게 되었다고 함
- 이 때, 커밋을 새로 추가하지 않고, 방금했던 커밋을 수정할 수 있음
- 이미 원격저장소에까지 push했더라도 가능함
- 파일 뿐 아니라 커밋 메시지 수정하는 것도 가능함

Amend 실습 (로컬저장소에만 올린 커밋)

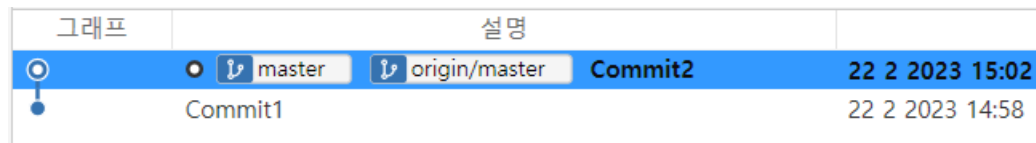
- 아래와 같이 커밋 두 개를 한 상황이라고 함
 - 로컬 폴더명: GitDirectory1
 - GitFile1.txt 파일을 생성함. 내용은 Contents1, Contents2
 - 커밋 메시지: Commit1, Commit2



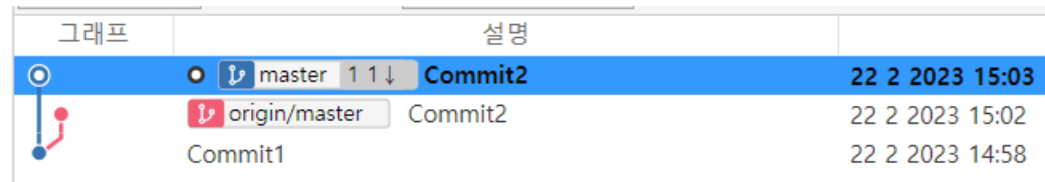
- 이 때, 사실은 Contents2가 아니라 Contents3라고 쓰고 싶었다고 함 . GitFile1.txt의 내용을 Contents3으로 수정함
- 이제 커밋을 하되 바로 커밋 버튼을 누르지 않고 우측 하단의 [커밋 옵션] 드롭다운 버튼을 누르고 [마지막 커밋 정정] 버튼을 누른다. 영어로는 Amend last commit이다. 이는 마지막 커밋을 수정한다는 의미이다. 그 후 커밋을 누름
- 그러면 그래프 모양은 동일하되 Commit2가 내용을 Contents3으로 수정한 것이 됨

Amend 실습 (push까지 한 커밋)

- 아래와 같이 커밋 두 개를 한 상황이라고 함. 여기에 push까지 함
 - 로컬 폴더명: GitDirectory1
 - GitFile1.txt 파일을 생성함. 내용은 Contents1, Contents2
 - 커밋 메시지: Commit1, Commit2
 - 원격저장소 이름: GitRepository1

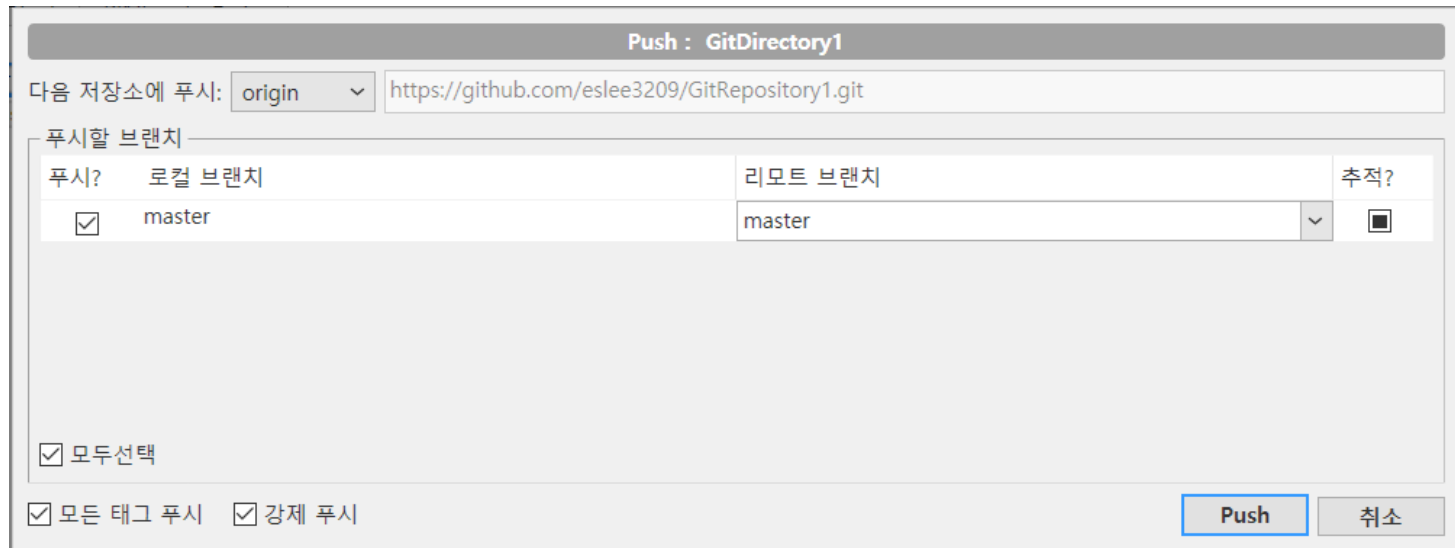


- 원격저장소에까지 push한 커밋을 수정하기 위해 GitFile1.txt의 내용을 Contents3으로 바꾸고 커밋을 하되 아까처럼 amend를 함. 그러면 아래와 같이 갈래가 생김



Amend 실습 (push까지 한 커밋)

- 이제 push를 하되 [강제 푸시]를 체크해야함. 원격저장소에 강제로 덮어쓰우는 푸시임



Push : GitDirectory1

다음 저장소에 푸시: origin https://github.com/eslee3209/GitRepository1.git

푸시할 브랜치






푸시?	로컬 브랜치	리모트 브랜치	추적?
<input checked="" type="checkbox"/>	master	master	<input type="checkbox"/>

☒ 모두선택

☒ 모든 태그 푸시 ☒ 강제 푸시

Push 취소

- 그러면 그래프가 깔끔해짐

그래프	설명	
	  master  origin/master Commit2	22 2 2023 15:03
	Commit1	22 2 2023 14:58

