개발 입문 스터디

3주차

출석체크

Announcement

복습

Issue

Repository에서 작업 계획, 토론 및 추적을 위해 활용

Branch

- 기존 브랜치에서 분기되어 생성되는 별도의 작업 공간
- fork와 달리 같은 Repository에 생성됨
- 한 번 Merge한 브랜치를 재활용 하는 것은 지양

Branch

현재 브랜치 확인하기

```
$ git branch
```

모든 브랜치 확인하기

```
$ git branch -a
```

Branch 생성/삭제

브랜치 생성하기

$ git branch “<브랜치 이름>”

브랜치 삭제하기

$ git branch -D “<브랜치 이름>”

Branch - checkout

브랜치 이동하기

$ git checkout “<브랜치 이름>”

브랜치 생성 후 이동하기

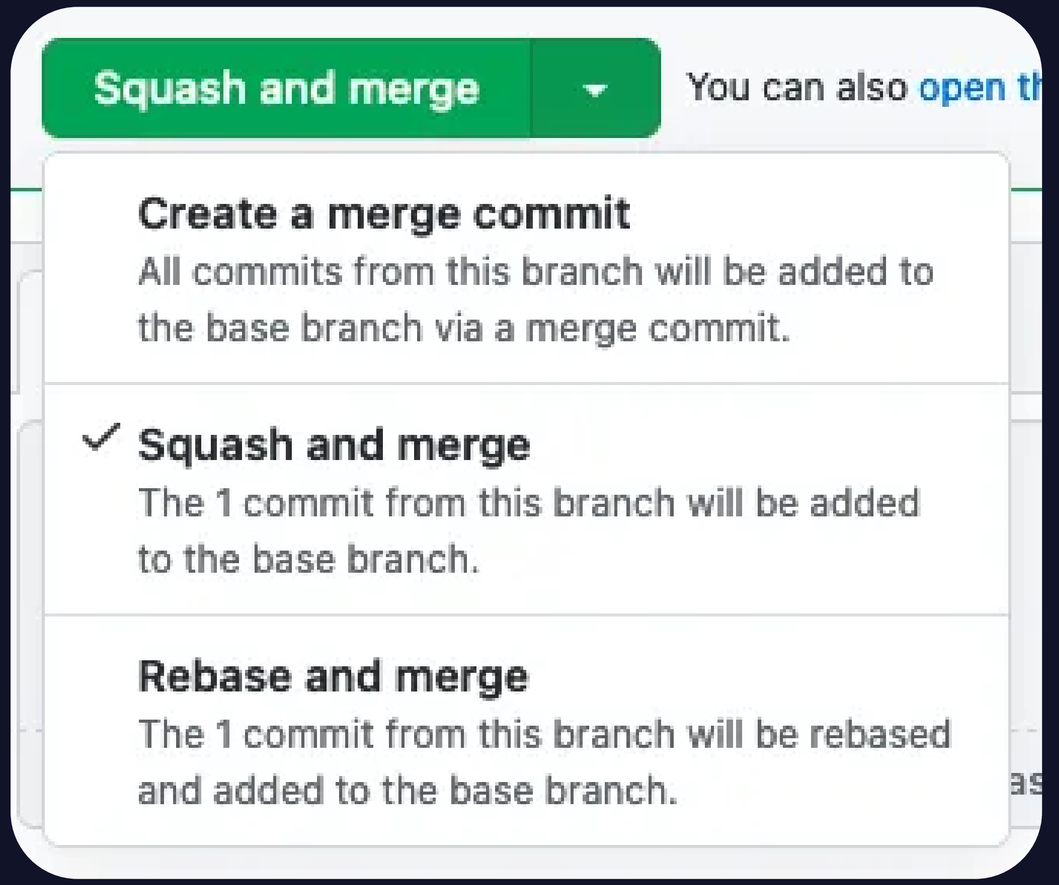
$ git checkout -b “<브랜치 이름>”

Pull Request

- 분기된 Branch를 다시 병합하기 위한 절차
- 새로운 변경을 제안하거나 병합 시 발생하는 충돌을 해결
- Merge에 앞서 코드 리뷰

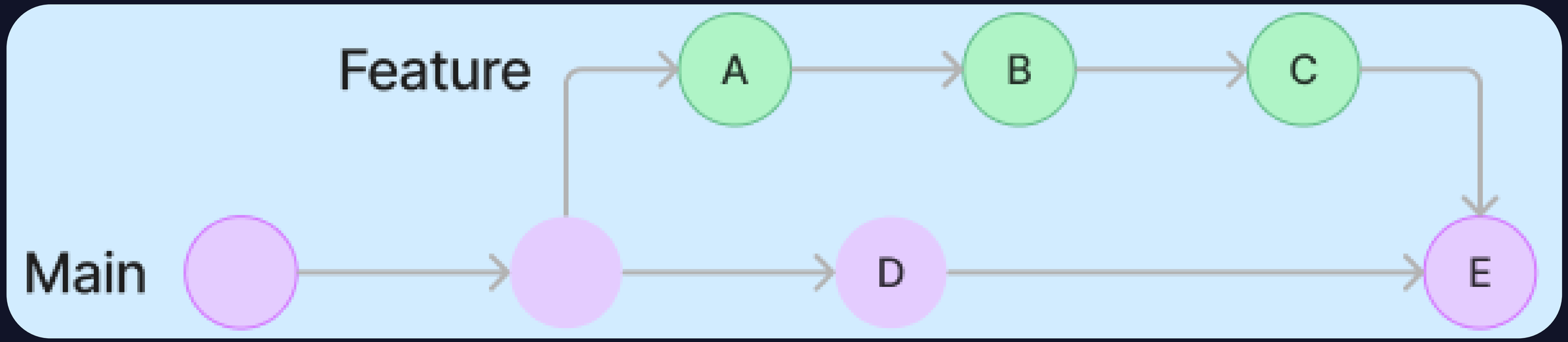
Merge

Merge에는 3가지 옵션이 존재



Merge Commit

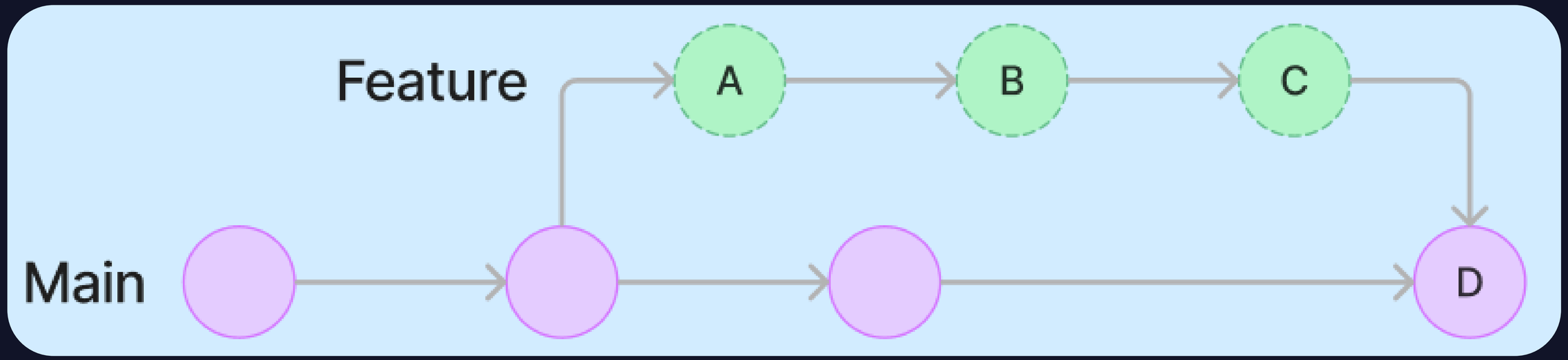
- 두 브랜치를 공통 부모로 하는 새로운 commit ‘E’를 만듦
- A, B, C의 커밋이 그대로 main 브랜치로 병합



Squash and Merge

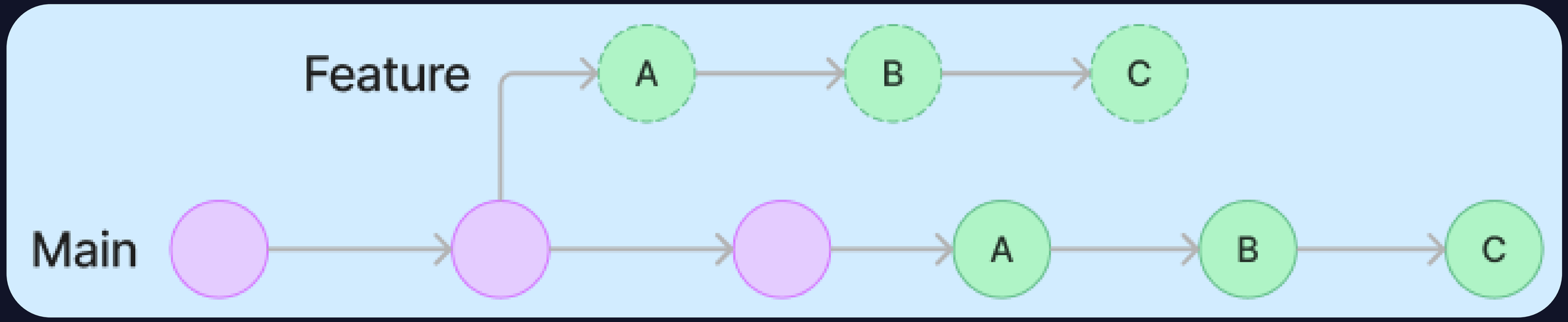
- A, B, C의 커밋을 ’squash’

⇒ 하나의 커밋으로 main 브랜치로 병합



Rebase and Merge

- A, B, C 커밋의 base를 재설정
- 모두 새로운 커밋으로 변경



지난 과제

오늘의 토픽

- 사전 지식
  - git log
    - Commit Id
    - HEAD
  - git status
- commit 되돌리기
  - amend
  - reset
  - revert

사전 지식

git log

- commit 기록을 최신 순으로 확인
- --oneline 옵션을 사용하면 각 커밋을 한 줄에 요약

```
aee7ca2 (HEAD -> develop, origin/develop, origin/HEAD) hotfix: spotless 적용 (#285)
e462002 hotfix: 학과 쿼리 메서드 수정 (#284)
1f39f54 feat: 2차 모집 기간 마감 시 LandingStatus 조건 추가 (#278)
24a51dd refactor: 발생 가능성 있는 NPE를 제거 (#170)
a67923c fix: CI 워크플로우에서 Gradle 빌드가 수행되는 문제 해결 (#276)
2319f1f refactor: 쿼리 메서드 이름 개선 및 연관 로직 주석 추가 (#274)
```

Commit Id

- commit의 식별을 위해 사용하는 40자 길이의 16진수
- 중복 확률은 2의 80제곱 분의 1
- SHA-1 해시 함수를 사용

```
commit aee7ca2135a2501fbc8db90ae2dcc77dde370f0b (HEAD -> develop, origin/develop, origin/HEAD)
Author: Cho Sangwook <82208159+Sangwook02@users.noreply.github.com>
Date:   Fri Mar 8 22:28:50 2024 +0900

    hotfix: spotless 적용 (#285)

    * hotfix: 이메일 정규식에 언더스코어를 허용하도록 수정 (#205)
```

HEAD

- HEAD는 현재 작업 중인 위치를 가리킨다
- 현재 작업중인 브랜치의 가장 최근 commit
- 다음 commit의 base가 되는 commit
- 새로운 commit이 생기면 HEAD도 변경

```
aee7ca2 (HEAD -> develop, origin/develop, origin/HEAD) hotfix: spotless 적용 (#285)
e462002 hotfix: 학과 쿼리 메서드 수정 (#284)
1f39f54 feat: 2차 모집 기간 마감 시 LandingStatus 조건 추가 (#278)
24a51dd refactor: 발생 가능성 있는 NPE를 제거 (#170)
a67923c fix: CI 워크플로우에서 Gradle 빌드가 수행되는 문제 해결 (#276)
2319f1f refactor: 쿼리 메서드 이름 개선 및 연관 로직 주석 추가 (#274)
```

git status

세 가지 상태에 따라 파일을
분류하여 확인

commit --amend

- 마지막 commit의 내용에 변경이 있을 때 사용
- 완전히 새로운 commit으로 대체
  - commit id가 바뀜
- vim으로 진입하여 commit 메시지를 수정하게 됨

commit --amend

‘-m’ option으로 vim 진입 없이 commit 메시지 수정하기
$ git commit --amend -m “커밋 메시지”

‘--no-edit’ option으로 commit 메시지 수정 없이 commit 수정
$ git commit --amend --no-edit

🚨 commit --amend 주의사항

다른 사람이 작업 기반으로 삼고 있는 commit은
amend하면 안 됩니다!

reset

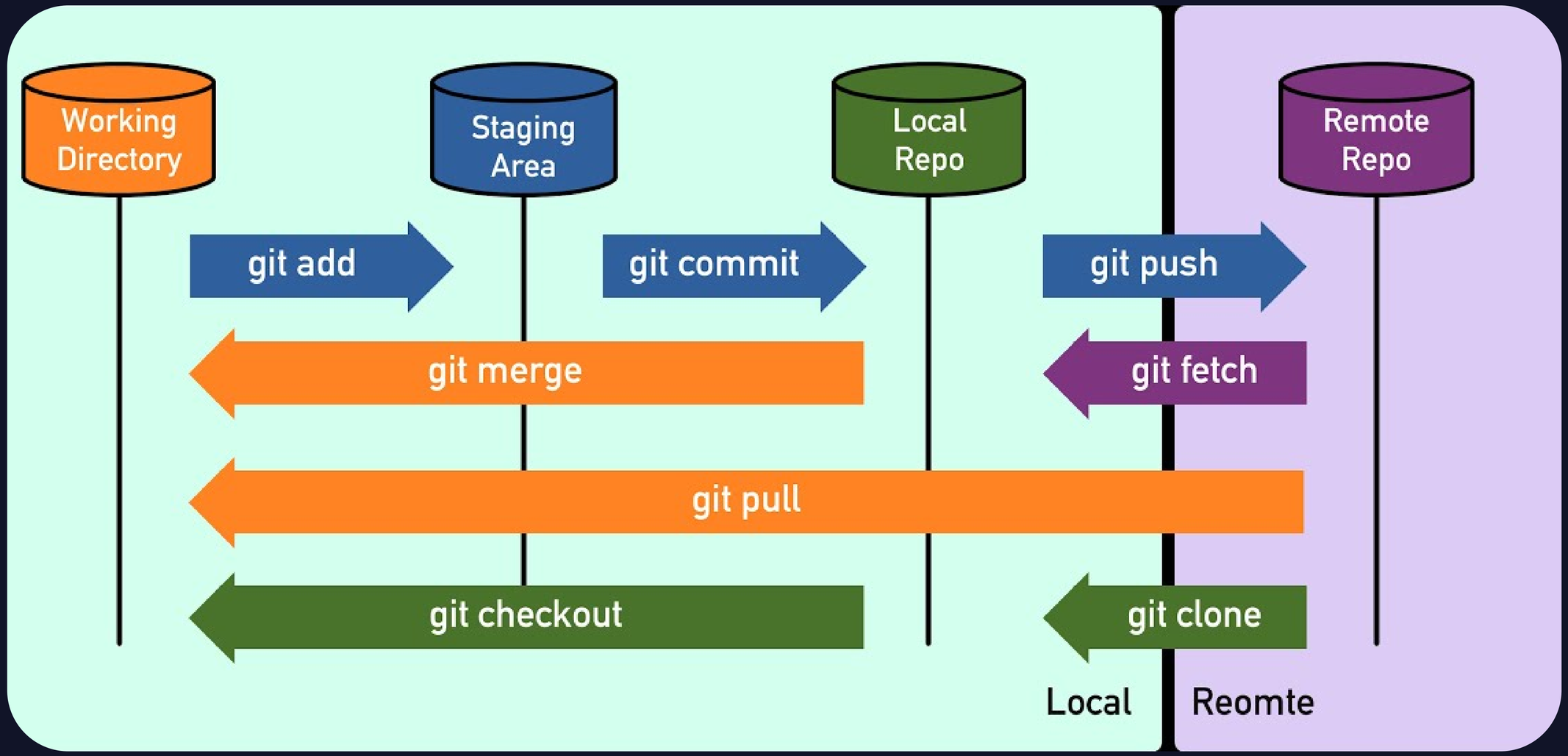
- commit을 제거하는 데 사용
- 3가지 옵션 존재
  - soft, mixed(default), hard

reset

돌아갈 commit의 id를 사용

$ git reset ‘--option’ “<commit id>”

ex) $ git reset --soft a1s2d3f


Working Directory
Staging Area
Local Repo
Remote Repo
git add
git commit
git push
git merge
git fetch
git pull
git checkout
git clone
Local
Reomte

reset --soft

- 커밋만 취소
- 변경 사항이 Staging Area로 돌아감

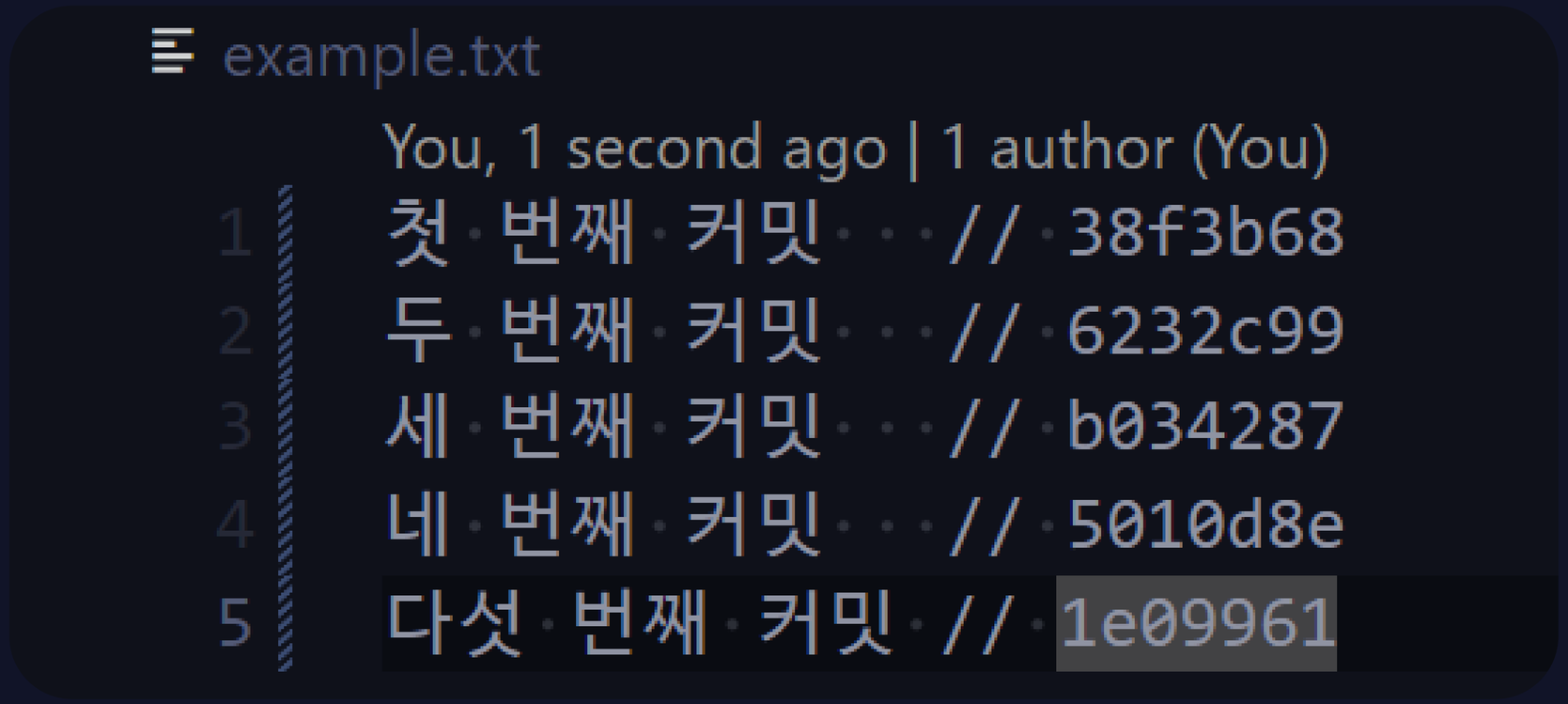
reset --mixed

- 커밋을 취소
- 변경 사항을 working directory로 돌아감

reset --hard

- 커밋을 취소
- 변경 사항을 모두 제거하고 이전 커밋으로 돌아감

example - Situation



example - Situation

```
1e09961 (HEAD -> main) docs: fifth commit
5010d8e docs: fourth commit
b034287 docs: third commit
6232c99 docs: second commit
38f3b68 docs: first commit
```

git log --oneline

Case 1) ‘git reset --soft b034287’의 결과는?

네 번째 커밋
다섯 번째 커밋

이라는 내용이 Staging Area로 돌아감

```
PS C:\Users\chosw\Desktop\개발_입문_스터디\3주차\example> git status
On branch soft
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   example.txt
```

Case 2) ‘git reset --mixed b034287’의 결과는?

네 번째 커밋
다섯 번째 커밋

이라는 내용이 Working Directory로 돌아감

```
PS C:\Users\chosw\Desktop\개발_입문_스터디\3주차\example> git status
On branch test
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   example.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

Case 3) ‘git reset --mixed b034287’의 결과는?

네 번째 커밋
다섯 번째 커밋

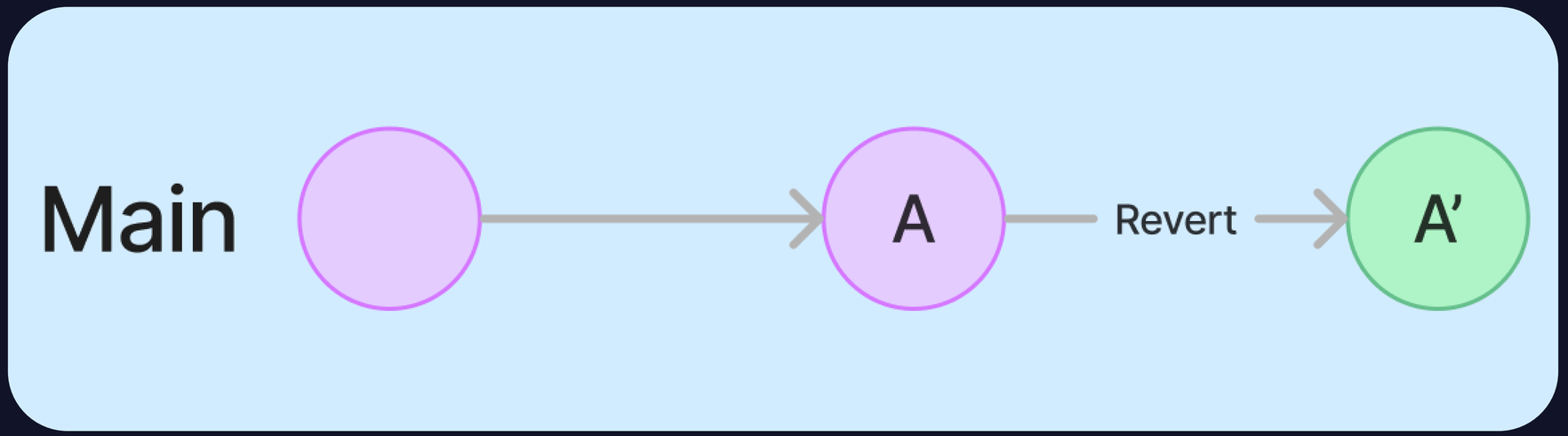
이라는 내용이 완전히 사라짐

```
PS C:\Users\chosw\Desktop\개발_입문_스터디\3주차\example> git status
On branch hard
nothing to commit, working tree clean
```

revert

- commit을 제거하지 않고 되돌림
- 되돌리기 위한 새로운 commit이 생성됨
- --edit 옵션이 default

  ex) $ git revert a1s2d3f



revert --no-edit

- 편집기 진입 없이 바로 revert 가능

$ git revert --no-edit “<commit id>”

```
b506241 (HEAD -> revert) Revert "docs: fifth commit"
1e09961 (main) docs: fifth commit
5010d8e docs: fourth commit
b034287 docs: third commit
6232c99 docs: second commit
38f3b68 docs: first commit
```

revert --no-commit

- 직접 commit 하지 않고, revert 내용을 Staging Area에 올림

```
b506241 (HEAD -> revert) Revert "docs: fifth commit"
1e09961 (main) docs: fifth commit
5010d8e docs: fourth commit
b034287 docs: third commit
6232c99 docs: second commit
38f3b68 docs: first commit
```

reset vs revert

- reset은 commit을 삭제하므로 위험
- commit을 공유하는 다른 브랜치에도 영향을 줄 수 있음
- commit을 삭제하기보다 생성하여 되돌리는 revert가 안전

Summary

Assignment

고생하셨습니다!