git

1. 준비

- git 설치 확인

```
$ git --version
```

- git 사용자 설정 :

```
$ git config --global user.email "email"
$ git config --global user.name "name"
```

- 연습 디렉토리 만들고 이동하기

```
$ mkdir gitPractice && cd gitPractice
```

- git 저장소 초기화 (구글 클라우드 콘솔에서 연습할 때 필요)

init

```
$ git branch -m main
```

- 필요시 초기 브랜치 이름을 master 에서 main 등 다른 이름으로 변경

```
$ git branch -m main
```

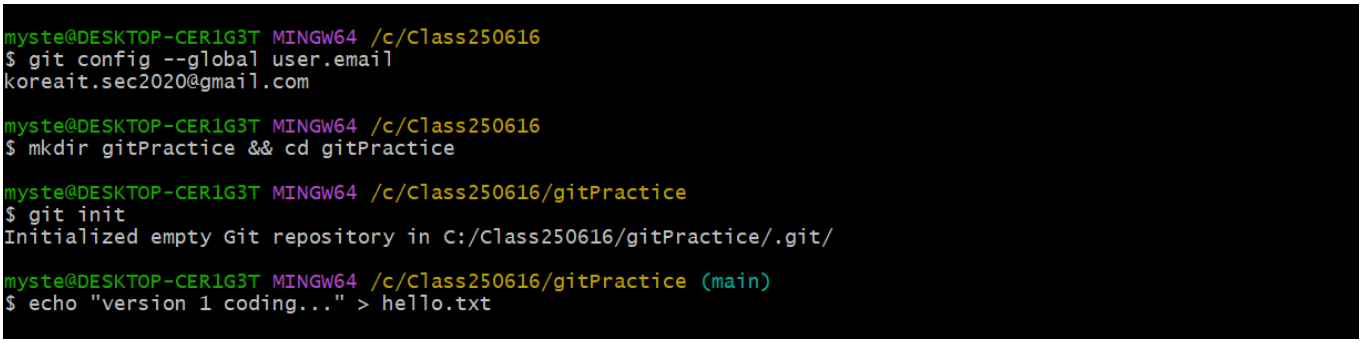
- 연습 파일 hello.txt 만들기

```
$ echo "version 1 coding.." > hello.txt
```

- 파일 생성 및 내용 확인

```
$ cat hello.txt
$ ls hello.txt
```

[git bash 에서 실행한 화면]



2. 스테이징

- 파일 목록 확인하기
- hello.txt 스테이징 하기
- 스테이징 상태 확인하기
- 언스테이징 하기
- 스테이징 상태 확인하기
- 다음 실습을 위해 다시 스테이징 하기 : `git add hello.txt`

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ ls
hello.txt

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git add hello.txt
warning: in the working copy of 'hello.txt', LF will be replaced by CRLF the next time Git touches it

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ cat hello.txt
version 1 coding...

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   hello.txt


myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git reset hello.txt

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        hello.txt

nothing added to commit but untracked files present (use "git add" to track)

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$
```

3. 커밋

`$ git add hello.txt` 다시 실행 후

- 커밋하기
- 커밋 로그 확인하기
- 커밋 로그 (한줄 확인)

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git commit -m "start hello"
[main (root-commit) 6f9706c] start hello
 1 file changed, 1 insertion(+)
 create mode 100644 hello.txt

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git log
commit 6f9706cf3aadd75923a9c6ad241cd6ec55b7c388 (HEAD -> main)
Author: kimsohee-around <koreait.sec2020@gmail.com>
Date:   Sat Jul 19 15:09:13 2025 +0900

    start hello

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git log --oneline
6f9706c (HEAD -> main) start hello

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$
```

4. 브랜치 만들기 (1)

- feature/login 이름 브랜치 만들기
- hello.txt 에 내용 추가하기
- 커밋하기 (오류)

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git checkout -b feature/login
Switched to a new branch 'feature/login'

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/login)
$ ls
hello.txt

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/login)
$ echo "login coding...." >> hello.txt

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/login)
$ cat hello.txt
version 1 coding...
login coding....

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/login)
$ git commit -m "feature login"
On branch feature/login
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:   hello.txt

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

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/login)
$
```

5. 4번 계속

- 스테이징하기
- 커밋하기
- 브랜치 main 으로 변경하기
- 브랜치 main 의 hello.txt 파일 내용 확인하기

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/login)
$ git add hello.txt
warning: in the working copy of 'hello.txt', LF will be replaced by CRLF the next time Git touches it

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/login)
$ git commit -m "feature login"
[feature/login 466615d] feature login
 1 file changed, 1 insertion(+)

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/login)
$ git checkout main
Switched to branch 'main'

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ cat hello.txt
version 1 coding...
```

6. `git switch` 명령어 - 브랜치 만들기 (2)

- feature/logout 브랜치 만들기

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git switch
fatal: missing branch or commit argument

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git switch -h
usage: git switch [<options>] [<branch>]

    -c, --[no-]create <branch>
                          create and switch to a new branch
    -C, --[no-]force-create <branch>
                          create/reset and switch to a branch
    --[no-]guess          second guess 'git switch <no-such-branch>'
    --[no-]discard-changes
                          throw away local modifications
    -q, --[no-]quiet      suppress progress reporting
    --[no-]recurse-submodules[=<checkout>]
                          control recursive updating of submodules
    --[no-]progress       force progress reporting
    -m, --[no-]merge      perform a 3-way merge with the new branch
    --[no-]conflict <style>
                          conflict style (merge, diff3, or zdiff3)
    -d, --[no-]detach     detach HEAD at named commit
    -t, --[no-]track[=(direct|inherit)]
                          set branch tracking configuration
    -f, --[no-]force      force checkout (throw away local modifications)
    --[no-]orphan <new-branch>
                          new unborn branch
    --[no-]overwrite-ignore
                          update ignored files (default)
    --[no-]ignore-other-worktrees
                          do not check if another worktree is holding the given ref


myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git switch -c feature/logout
Switched to a new branch 'feature/logout'

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/logout)
$
```

7. 새로운 브랜치

- hello.txt 에 내용 추가하기
- hello.txt 내용 확인하기
- 스테이징 하기 와 커밋하기

8. 병합하기 (1)

- 브랜치 main 으로 변경하기
- feature/login 브랜치와 병합하기
    - 병합 방식 : Fast-forward

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/logout)
$ echo "logout coding..." >> hello.txt

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/logout)
$ cat


myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/logout)
$ cat hello.txt
version 1 coding...
logout coding...

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/logout)
$ git add hello.txt
warning: in the working copy of 'hello.txt', LF will be replaced by CRLF the next time Git touches it

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/logout)
$ git commit -m "feature logout"
[feature/logout 5d0ad02] feature logout
 1 file changed, 1 insertion(+)

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (feature/logout)
$ git checkout main
Switched to branch 'main'

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git merge feature/login
Updating 6f9706c..466615d
Fast-forward
 hello.txt | 1 +
 1 file changed, 1 insertion(+)

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$
```

9. 파일 편집기(프로그램)

- nano
- vi

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ nano hello.txt

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ vi hello.txt

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ cat hello.txt
version 1 coding...
login coding....

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$
```

깃 히스토리 확인

```
git log --oneline --graph --all
```

nano

종료하기는 Ctrl + X

vi

종료하기는 명령모드 Esc 후에 q! (저장없이 종료) 명령어 입력하기

10. 병합 시도 하기 (2)

충돌 발생

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git merge feature/logout
Auto-merging hello.txt
CONFLICT (content): Merge conflict in hello.txt
Automatic merge failed; fix conflicts and then commit the result.
```

충돌 해결

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main|MERGING)
$ vi hello.txt
```

```
version 1 coding...
<<<<<<< HEAD
login coding....
=======
logout coding...
>>>>>>> feature/logout
~
~
~
~
~
~
~
~
~
~
~
~
~
hello.txt [dos] (15:25 19/07/2025)                                  1,1 All
"hello.txt" [dos] 6L, 104B
```

다음과 같이 편집 후 저장

```
version 1 coding...
login coding....
logout coding...
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
hello.txt [dos] (15:27 19/07/2025)                                  3,1 All
"hello.txt" [dos] 3L, 57B
```

명령모드 Esc 입력 후 wq 명령 입력

11. 병합 커밋 후 취소하기

- 스테이징하기
- 스테이징 후 커밋 전에 취소하기 : **`git reset --merge`**

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main|MERGING)
$ cat hello.txt
version 1 coding...
login coding....
logout coding...

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main|MERGING)
$ git add hello.txt

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main|MERGING)
$ git reset --merge

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ cat hello.txt
version 1 coding...
login coding....

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
```

12. 다시 병합 시도하기 (3)

- 충돌 편집 하지 않고 바로 취소하기 : **`git merge --abort`**

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git merge feature/logout
Auto-merging hello.txt
CONFLICT (content): Merge conflict in hello.txt
Automatic merge failed; fix conflicts and then commit the result.

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main|MERGING)
$ cat hello.txt
version 1 coding...
<<<<<<< HEAD
login coding....
=======
logout coding...
>>>>>>> feature/logout

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main|MERGING)
$ git merge --abort

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ cat hello.txt
version 1 coding...
login coding....

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$
```

13. 다시 병합 시도하기 (4)

- 병합 명령 실행하기
- nano 편집기로 충돌 해결하기
- 스테이징 하기
- 커밋하기

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git merge feature/logout
Auto-merging hello.txt
CONFLICT (content): Merge conflict in hello.txt
Automatic merge failed; fix conflicts and then commit the result.

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main|MERGING)
$ nano hello.txt

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main|MERGING)
$ nano hello.txt

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main|MERGING)
$ cat hello.txt
version 1 coding...
login coding....
logout coding...

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main|MERGING)
$ git add *

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main|MERGING)
$ git commit -m "merge logout"
[main 2939400] merge logout

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$
```

충돌 해결하기

```
  GNU nano 8.1                                hello.txt
version 1 coding...
<<<<<<< HEAD
login coding....
=======
logout coding...
>>>>>>> feature/logout

                    [ Read 6 lines (converted from DOS format) ]
^G Help       ^O Write Out  ^F Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo
```

병합 커밋 로그 확인하기

```
$ git log
commit 293940089efd0db50e4d5a3700ac90bc3c6cbb22 (HEAD -> main)
Merge: 466615d 5d0ad02
Author: kimsohee-around <koreait.sec2020@gmail.com>
Date:   Sat Jul 19 16:02:37 2025 +0900

    merge logout

commit 5d0ad02c5bb06914a6c91c86f4d4df80d3456dc9 (feature/logout)
Author: kimsohee-around <koreait.sec2020@gmail.com>
Date:   Sat Jul 19 15:19:29 2025 +0900

    feature logout

commit 466615d9d26c1f22117c2a0a5ec6683fff3df3a2 (feature/login)
Author: kimsohee-around <koreait.sec2020@gmail.com>
Date:   Sat Jul 19 15:13:53 2025 +0900

    feature login

commit 6f9706cf3aadd75923a9c6ad241cd6ec55b7c388
Author: kimsohee-around <koreait.sec2020@gmail.com>
Date:   Sat Jul 19 15:09:13 2025 +0900

    start hello

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$
```

14. 커밋 완료 후 병합 취소하기

- 커밋까지 완료하고 병합 취소하기 : **`git revert -m 1 병합해시값`**

```
myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git log --oneline
2939400 (HEAD -> main) merge logout
5d0ad02 (feature/logout) feature logout
466615d (feature/login) feature login
6f9706c start hello

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ git revert -m 1 2939400
[main 5d7732c] Revert "merge logout"
 1 file changed, 1 deletion(-)

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$ cat hello.txt
version 1 coding...
login coding....

myste@DESKTOP-CER1G3T MINGW64 /c/Class250616/gitPractice (main)
$
```

`-m`은 **"mainline" (**기준 부모 커밋**)** 을 선택하는 옵션입니다. 병합 커밋은 부모가 **2**개 있음:

`-m 1`: 첫 번째 부모 (대부분 `main` 또는 `master`)

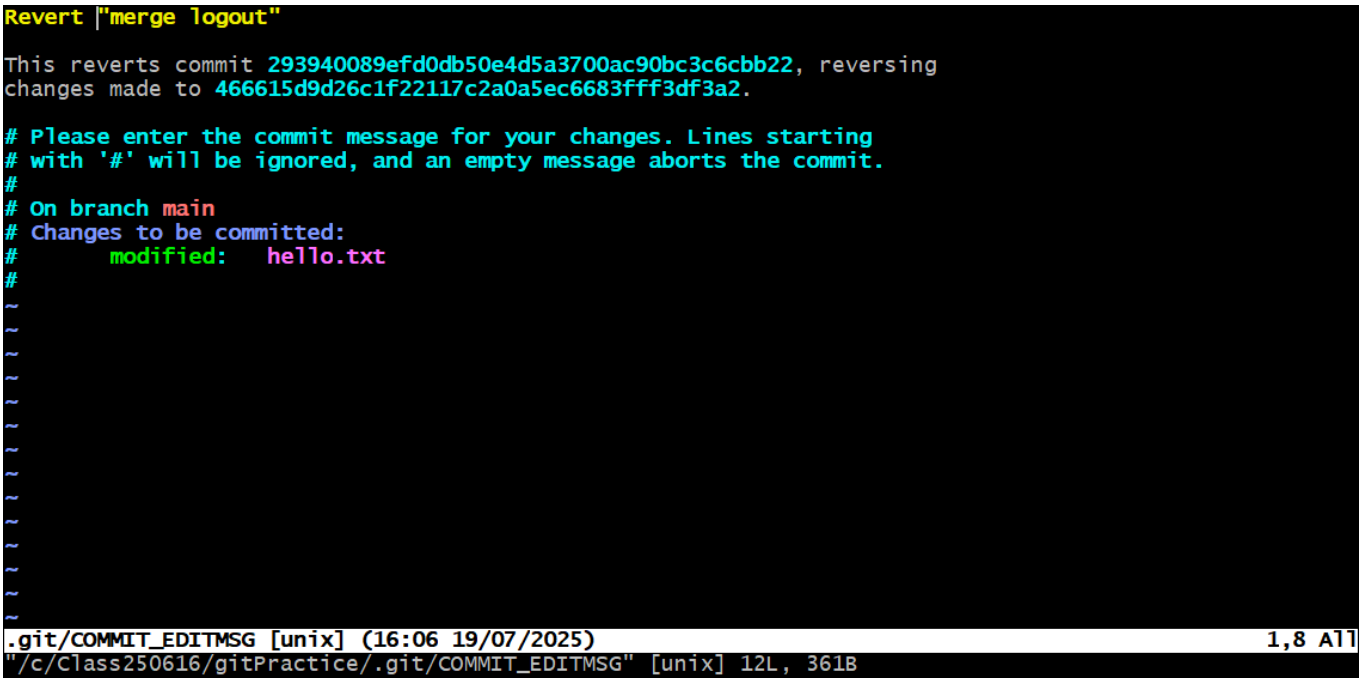
`-m 2`: 두 번째 부모 (예: feature 브랜치)

`main` 브랜치에서 `feature/logout` 를 병합한 경우 → `-m 1`

`feature/logout` 브랜치에서 `main`을 병합한 경우 → `-m 2`

병합 취소 메시지 편집기에서 수정하기

- 커밋 메시지 편집 하지 않을 때 **git revert -m 1 병합해시값 --no-edit**



일반적인 커밋 취소(❌PUSH 하기 전에만 사용. 협업 중에는 다른 팀원이 PULL 하기 전에 가능)

```
git reset --soft HEAD~1
```

커밋 메시지 수정

```
git commit --amend -m "커밋 수정"
```

[참고] 스테이징 또는 커밋한 파일 파일명 변경과 삭제

- **git mv** 또는 **git rm** 명령을 사용하여 파일 변경을 감지하도록 하며 이 때 git status 로 확인이 가능함.

```
# 기존 파일 만들기
echo "console.log('Hello');" > script.js
git add script.js
git commit -m "Add script.js"

# 파일 이름 변경
git mv script.js app.js
git commit -m "script.js to app.js 파일명 변경"

# 삭제
git rm app.js
git commit -m "Remove app.js 파일 삭제"
```