

Git & GitHub

◆ 브랜치

정수아

Contents

01 브랜치란?

02 초기 작업하기

03 브랜치 만들기

04 브랜치 정보 확인하기

05 브랜치 병합하기

06 정리하기



01

브랜치란?

브랜치란?

❖ 브랜치(Branch)

- 기준 브랜치로부터 독립적인 작업 공간을 만들어주는 기능
- 여러 개발자가 서로 다른 작업을 진행 시, 서로의 작업에 영향을 주지 않기 위해 필요
- 분기(branch)와 병합(merge) 기능

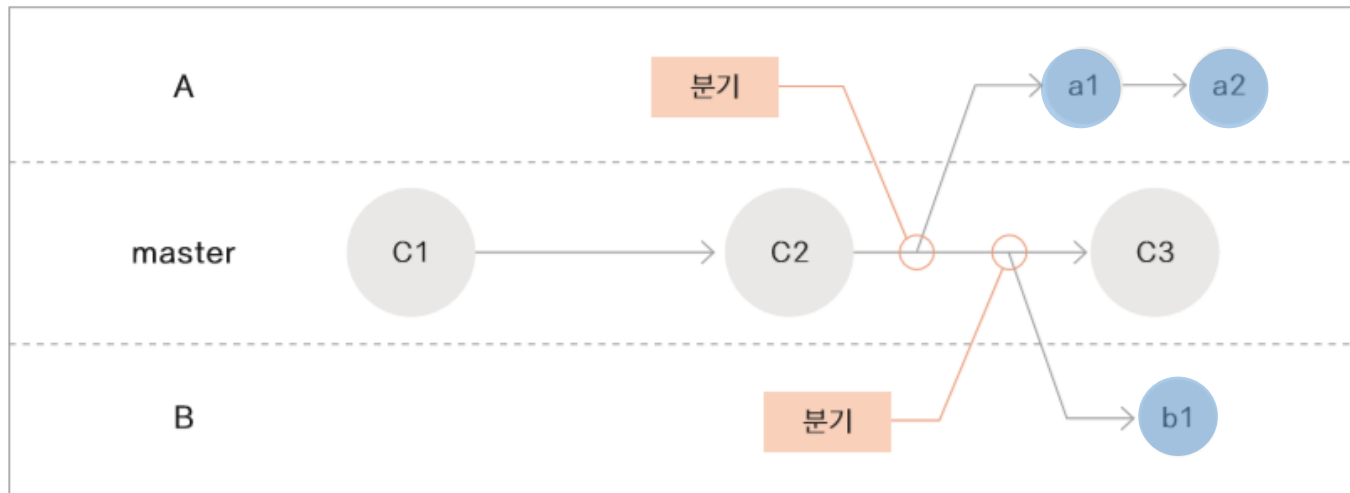
❖ main 또는 master 브랜치

- 깃으로 버전 관리를 시작하면 기본적으로 생성

브랜치 기능

❖ 분기(branch)

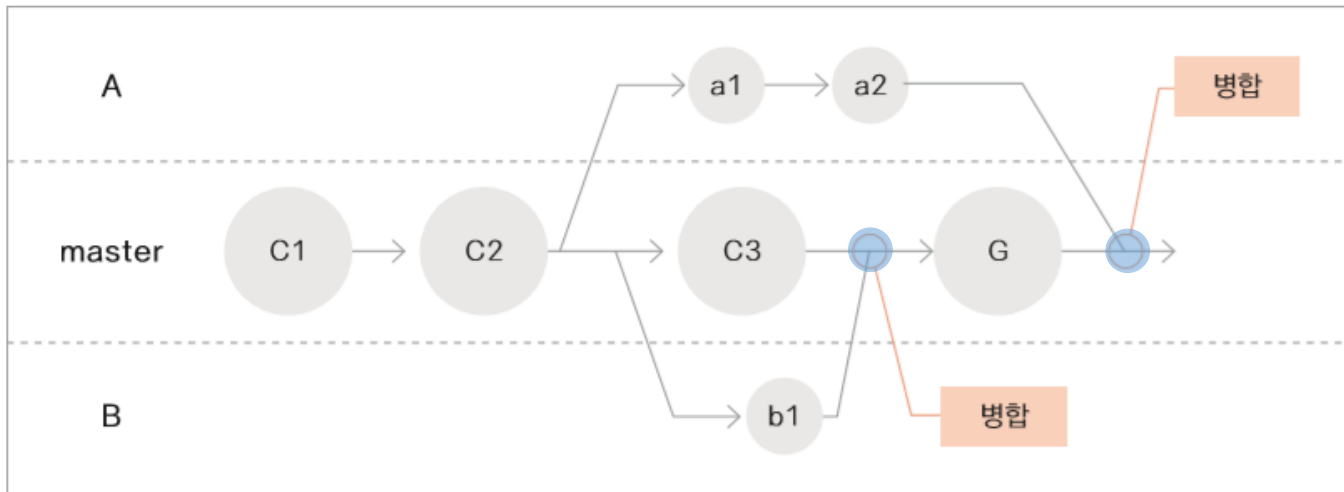
- master 브랜치에서 새 브랜치를 생성
- 기존에 저장한 파일을 master 브랜치에 유지하면서 기존 내용을 수정하거나 새로운 기능을 만들 수 있음



브랜치 기능

❖ 병합(merge)

- 새 브랜치에 있던 파일을 master 브랜치에 합침





02

초기 작업하기

초기 작업하기

❖ 디렉터리 생성

```
$ mkdir manual  
$ cd manual
```

❖ git 초기화

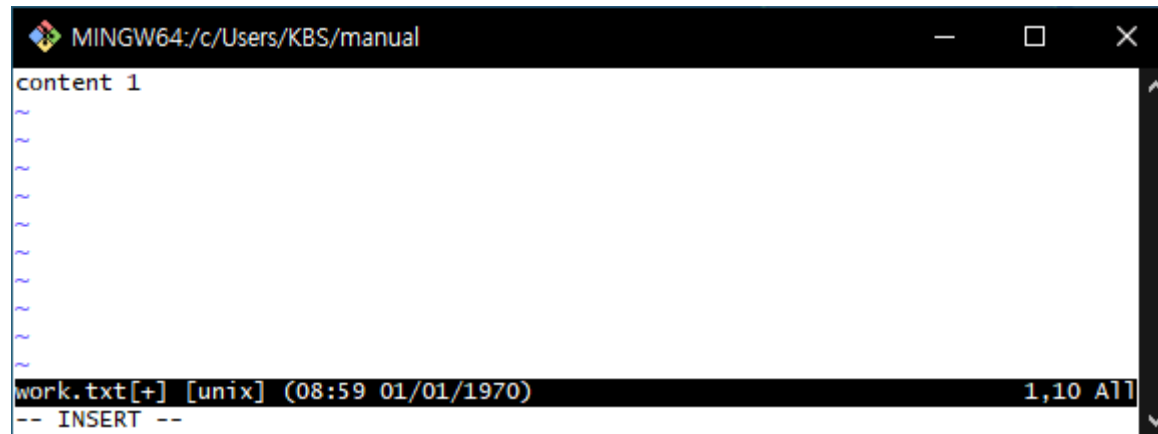
- 초기화 후 .git 디렉터리(숨김 폴더)가 생성

```
$ git init
```


초기 작업하기

❖ 파일 생성 후 내용 입력(content 1)

```
$ vim work.txt
```



초기 작업하기

❖ 스테이지에 올리기

```
$ git add work.txt
```

❖ 커밋하기

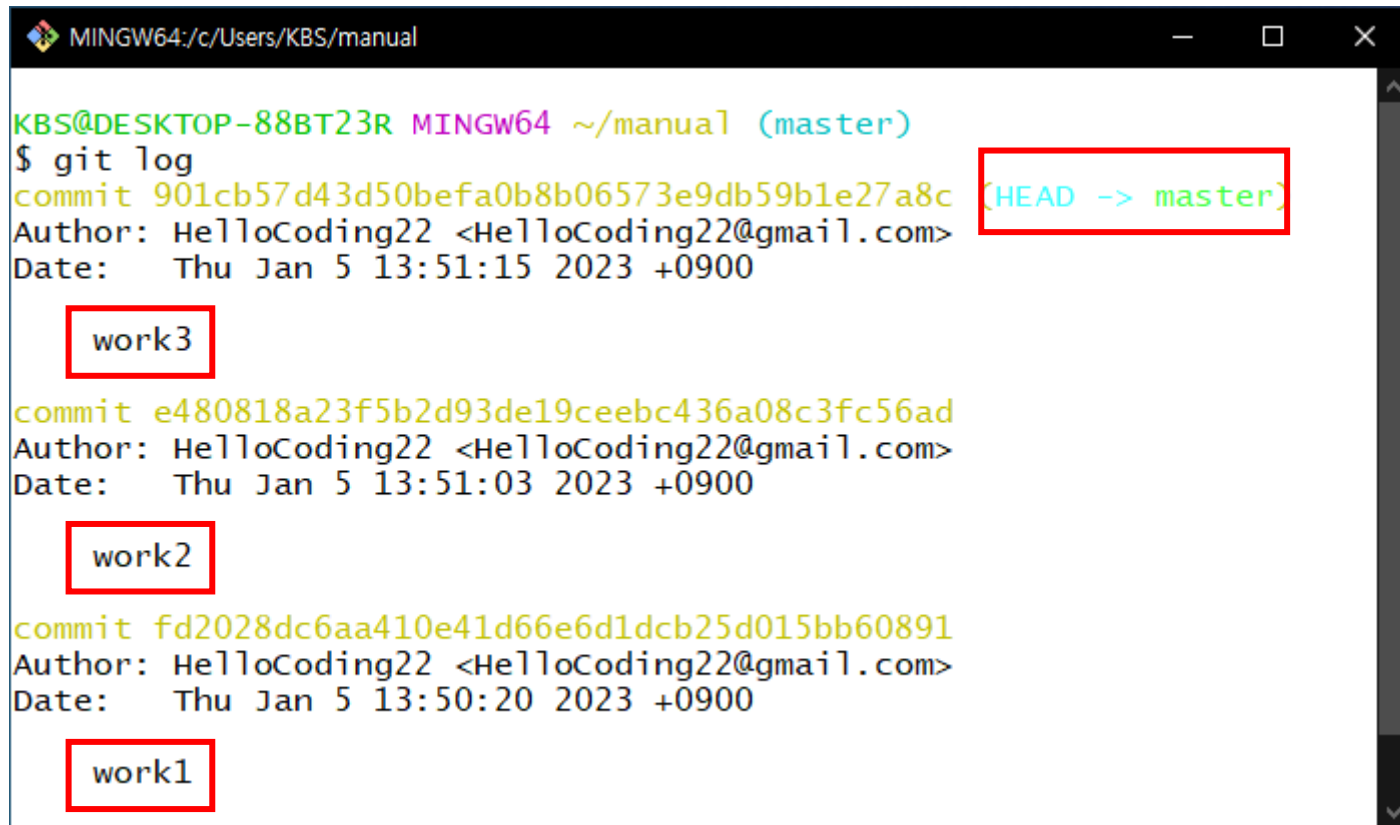
```
$ git commit -m "work1"
```

❖ 내용 입력 및 커밋하기 두 번 반복

초기 작업하기

❖ 커밋 내역 확인하기

```
$ git log
```



A screenshot of a Windows terminal window titled "MINGW64:/c/Users/KBS/manual". The terminal shows the output of the command `git log`. The output lists three commits in reverse chronological order. Each commit line is preceded by a label in a red box: "work3", "work2", and "work1". The first commit (901cb57d43d50befa0b8b06573e9db59b1e27a8c) is also followed by a red box containing the text "(HEAD -> master)".

```
MINGW64:/c/Users/KBS/manual
KBS@DESKTOP-88BT23R MINGW64 ~/manual (master)
$ git log
commit 901cb57d43d50befa0b8b06573e9db59b1e27a8c (HEAD -> master)
Author: HelloCoding22 <HelloCoding22@gmail.com>
Date: Thu Jan 5 13:51:15 2023 +0900
    work3

commit e480818a23f5b2d93de19ceebc436a08c3fc56ad
Author: HelloCoding22 <HelloCoding22@gmail.com>
Date: Thu Jan 5 13:51:03 2023 +0900
    work2

commit fd2028dc6aa410e41d66e6d1dcb25d015bb60891
Author: HelloCoding22 <HelloCoding22@gmail.com>
Date: Thu Jan 5 13:50:20 2023 +0900
    work1
```

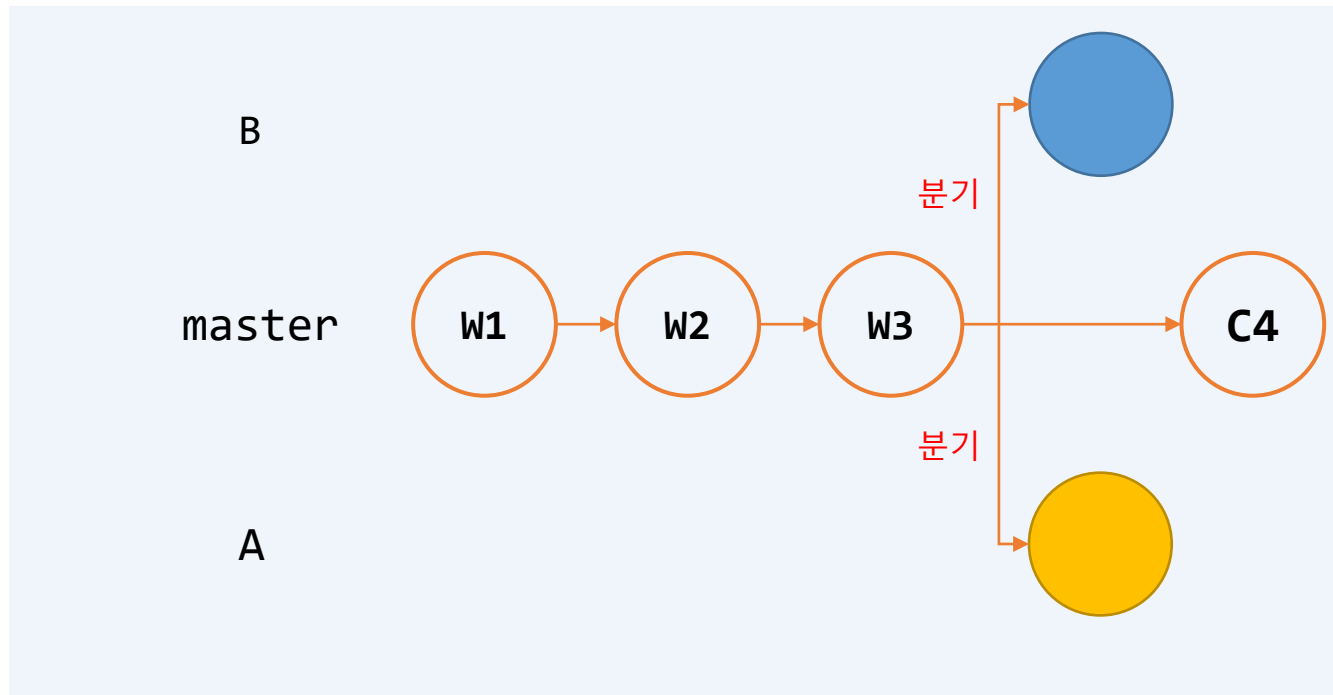


03

브랜치 만들기

브랜치 만들기

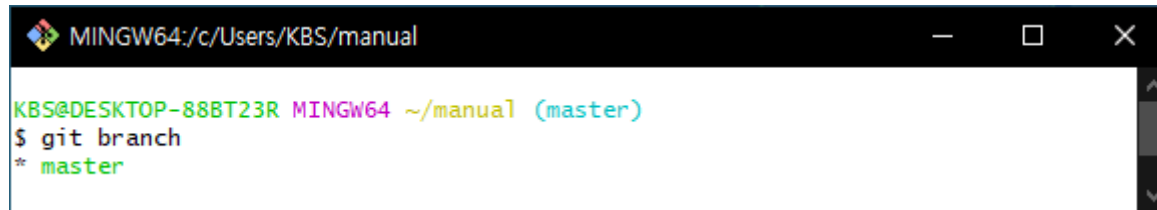
❖ 브랜치 분기



브랜치 만들기

❖ 브랜치 확인

```
$ git branch
```

A screenshot of a Windows terminal window titled 'MINGW64: c:/Users/KBS/manual'. The prompt is 'KBS@DESKTOP-88BT23R MINGW64 ~/manual (master)'. The command '\$ git branch' has been entered, and the output is '* master', indicating the current branch.

```
MINGW64: c:/Users/KBS/manual  
KBS@DESKTOP-88BT23R MINGW64 ~/manual (master)  
$ git branch  
* master
```

❖ 새 브랜치 생성

```
$ git branch 브랜치이름
```

브랜치 만들기

❖ A사의 브랜치 생성

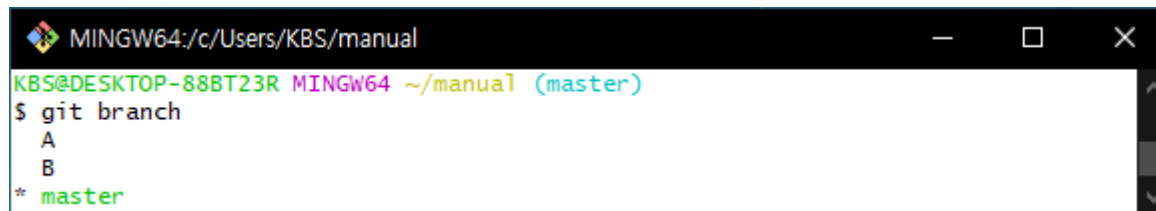
```
$ git branch A
```

❖ B사의 브랜치 생성

```
$ git branch B
```

❖ 생성된 브랜치 확인

```
$ git branch
```

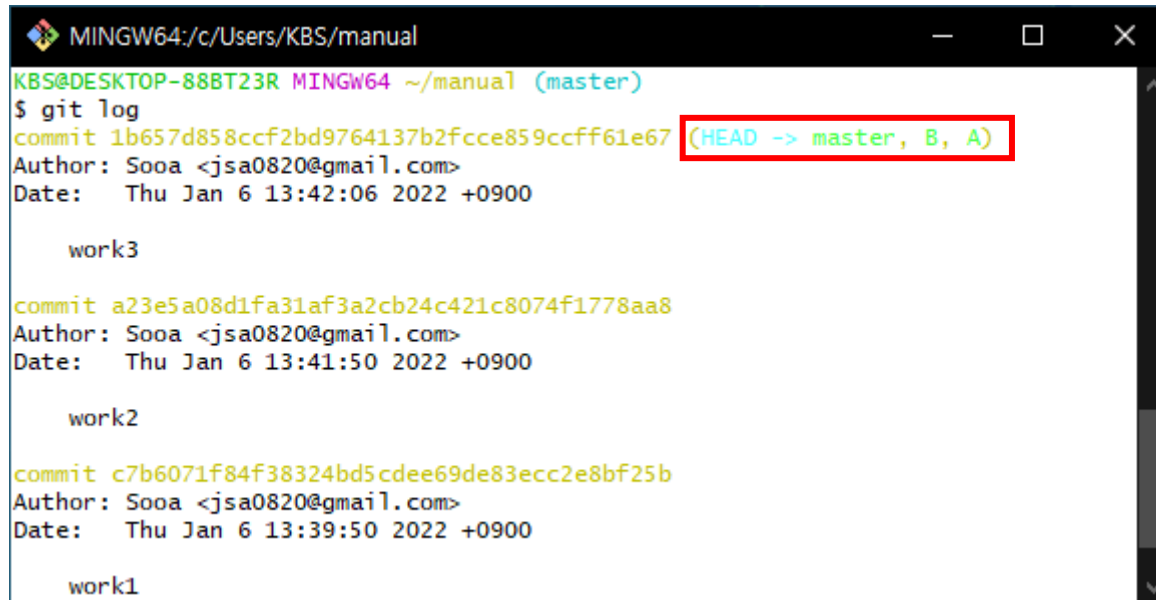


```
MINGW64:/c/Users/KBS/manual
KBS@DESKTOP-88BT23R MINGW64 ~/manual (master)
$ git branch
A
B
* master
```

브랜치 만들기

❖ 커밋 내역 확인하기

```
$ git log
```



```
MINGW64:/c/Users/KBS/manual
KBS@DESKTOP-88BT23R MINGW64 ~/manual (master)
$ git log
commit 1b657d858ccf2bd9764137b2fcce859ccff61e67 (HEAD -> master, B, A)
Author: Sooa <jsa0820@gmail.com>
Date: Thu Jan 6 13:42:06 2022 +0900

    work3

commit a23e5a08d1fa31af3a2cb24c421c8074f1778aa8
Author: Sooa <jsa0820@gmail.com>
Date: Thu Jan 6 13:41:50 2022 +0900

    work2

commit c7b6071f84f38324bd5cdee69de83ecc2e8bf25b
Author: Sooa <jsa0820@gmail.com>
Date: Thu Jan 6 13:39:50 2022 +0900

    work1
```


브랜치 만들기

❖ 새로운 내용 추가 및 커밋하기

- work.txt 파일에 content4 추가 후 커밋(C4)

```
$ git commit -am "master commit content 4"
```

브랜치 만들기

❖ 커밋 내역 확인하기

```
$ git log
```



The screenshot shows a terminal window titled 'MINGW64:/c/Users/KBS/manual'. The user is in the 'manual' directory on the 'master' branch. They run the command '\$ git log'. The output shows three commits. The first commit is highlighted with a red box around the hash and the branch name '(HEAD -> master)'. The second commit is highlighted with a red box around the text 'master commit content 4'. The third commit is highlighted with a red box around the text '(B, A)'. The output also shows the author 'HelloCoding22' and the date 'Thu Jan 5 13:57:05 2023 +0900' for the first commit, and 'Thu Jan 5 13:51:15 2023 +0900' for the second commit. The third commit is partially visible at the bottom.

```
MINGW64:/c/Users/KBS/manual
KBS@DESKTOP-88BT23R MINGW64 ~/manual (master)
$ git log
commit 19caa7e78e79f5089a456b33495f689cd660f975 (HEAD -> master)
Author: HelloCoding22 <HelloCoding22@gmail.com>
Date: Thu Jan 5 13:57:05 2023 +0900

    master commit content 4

commit 901cb57d43d50befa0b8b06573e9db59b1e27a8c (B, A)
Author: HelloCoding22 <HelloCoding22@gmail.com>
Date: Thu Jan 5 13:51:15 2023 +0900

    work3

commit e480818a23f5b2d93de19ceebc436a08c3fc56ad
Author: HelloCoding22 <HelloCoding22@gmail.com>
Date: Thu Jan 5 13:51:03 2023 +0900

    work2

commit fd2028dc6aa410e41d66e6d1dcb25d015bb60891
```

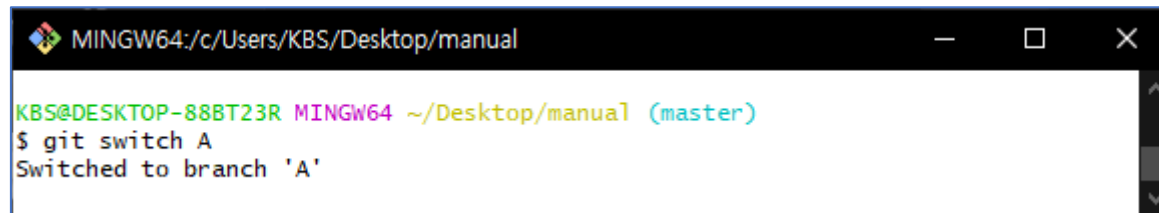
브랜치 변경 후 작업하기

❖ 브랜치 이동하기

```
$ git switch 브랜치이름
```

❖ A 브랜치로 이동하기

```
$ git switch A
```



```
MINGW64:/c/Users/KBS/Desktop/manual  
KBS@DESKTOP-88BT23R MINGW64 ~/Desktop/manual (master)  
$ git switch A  
Switched to branch 'A'
```

브랜치 변경 후 작업하기

❖ 커밋 내역 확인하기 - 현재 브랜치 : A

```
$ git log
```



A screenshot of a Windows terminal window titled "MINGW64:/c/Users/KBS/manual". The terminal shows the output of the command "git log". The output lists three commits. The first commit, with hash "901cb57d43d50befa0b8b06573e9db59b1e27a8c", is highlighted with a red box around the text "(HEAD -> A, B)". Below this commit, the word "work3" is also enclosed in a red box. The second commit, with hash "e480818a23f5b2d93de19ceebc436a08c3fc56ad", is followed by the word "work2". The third commit, with hash "fd2028dc6aa410e41d66e6d1dcb25d015bb60891", is followed by the word "work1". The terminal window has standard Windows window controls (minimize, maximize, close) in the top right corner.

```
MINGW64:/c/Users/KBS/manual
KBS@DESKTOP-88BT23R MINGW64 ~/manual (A)
$ git log
commit 901cb57d43d50befa0b8b06573e9db59b1e27a8c (HEAD -> A, B)
Author: HelloCoding22 <HelloCoding22@gmail.com>
Date: Thu Jan 5 13:51:15 2023 +0900

    work3

commit e480818a23f5b2d93de19ceebc436a08c3fc56ad
Author: HelloCoding22 <HelloCoding22@gmail.com>
Date: Thu Jan 5 13:51:03 2023 +0900

    work2

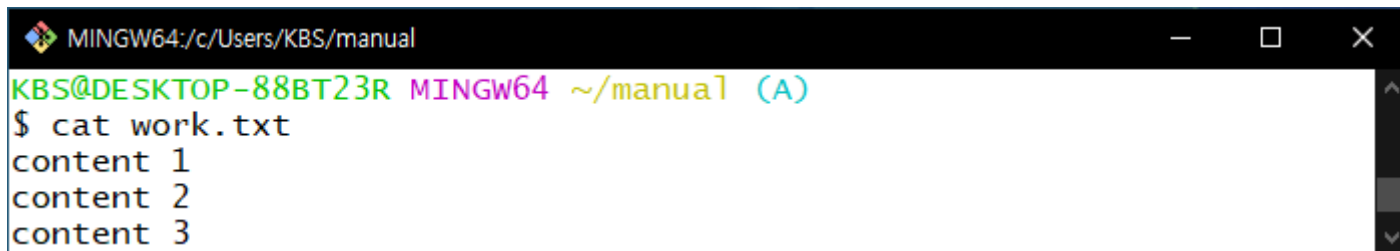
commit fd2028dc6aa410e41d66e6d1dcb25d015bb60891
Author: HelloCoding22 <HelloCoding22@gmail.com>
Date: Thu Jan 5 13:50:20 2023 +0900

    work1
```

브랜치 변경 후 작업하기

❖ **work.txt** 파일 확인하기 - 현재 브랜치 : A

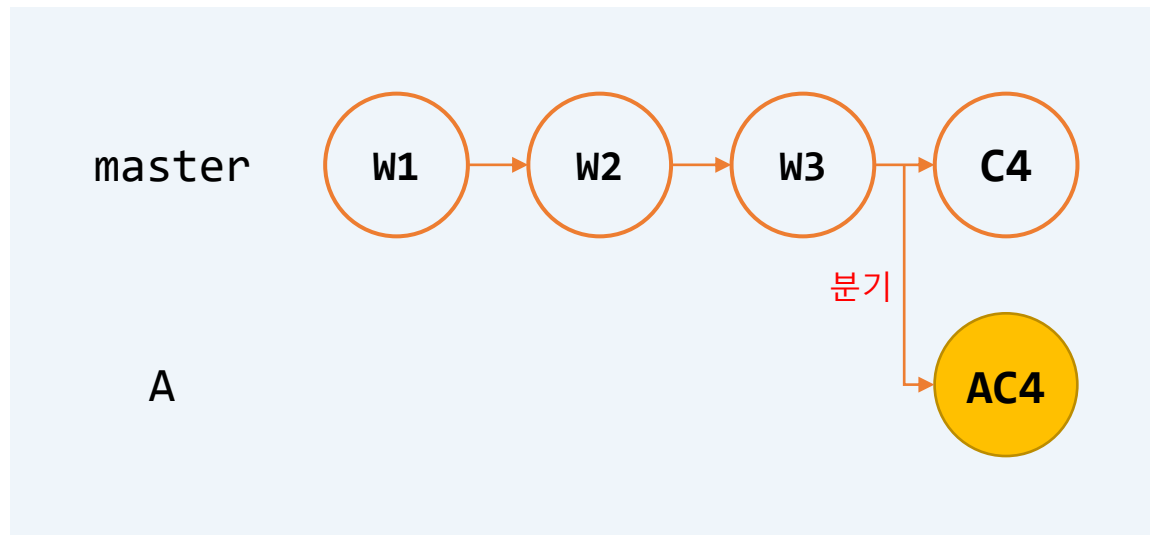
```
$ cat work.txt
```

A screenshot of a Windows terminal window titled 'MINGW64: c:/Users/KBS/manual'. The prompt is 'KBS@DESKTOP-88BT23R MINGW64 ~/manual (A)'. The command '\$ cat work.txt' has been entered, and the output is displayed as three lines: 'content 1', 'content 2', and 'content 3'.

```
MINGW64: c:/Users/KBS/manual
KBS@DESKTOP-88BT23R MINGW64 ~/manual (A)
$ cat work.txt
content 1
content 2
content 3
```

브랜치 변경 후 작업하기

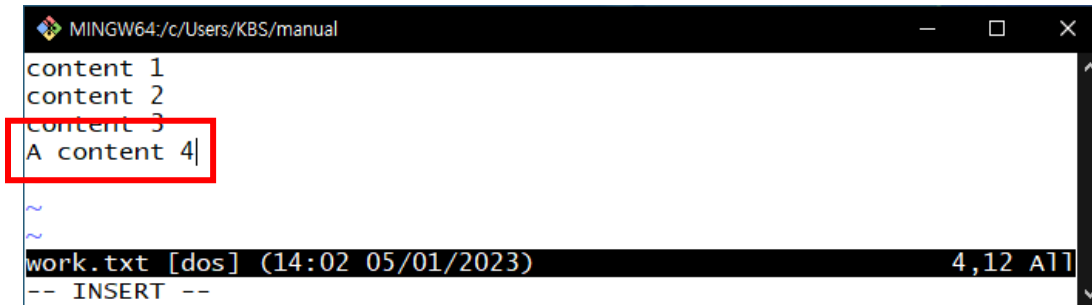
❖ 새로운 커밋 생성하기 - 현재 브랜치 : A



브랜치 변경 후 작업하기

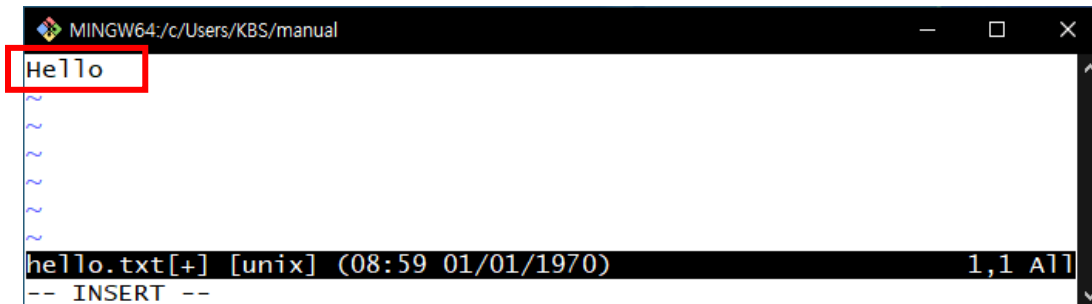
- ❖ A 브랜치의 work.txt 파일에 내용(A content 4) 추가

```
$ vim work.txt
```



```
MINGW64:/c/Users/KBS/manual
content 1
content 2
content 3
A content 4|
~
~
~
work.txt [dos] (14:02 05/01/2023) 4,12 All
-- INSERT --
```

- ❖ 새 파일(hello.txt) 생성 및 내용(Hello) 추가



```
MINGW64:/c/Users/KBS/manual
Hello
~
~
~
hello.txt[+] [unix] (08:59 01/01/1970) 1,1 All
-- INSERT --
```

브랜치 변경 후 작업하기

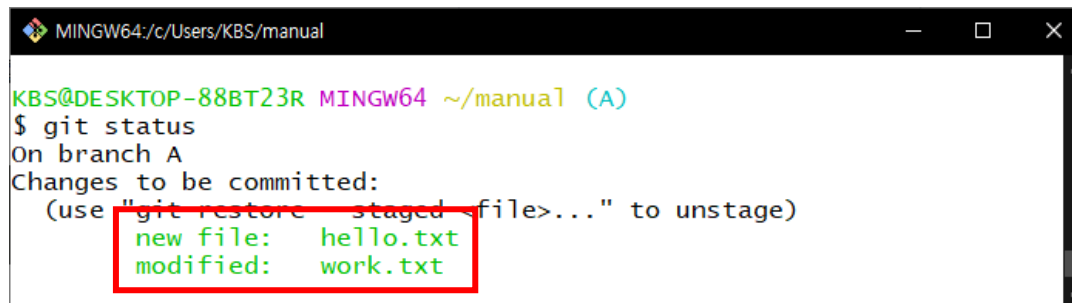
❖ 수정된 파일을 스테이지에 올리기

```
$ git add work.txt  
$ git add hello.txt
```

또는

```
$ git add .
```

- git add 명령어 뒤에 마침표(.)를 붙이면 현재 저장소에서 수정된 파일이 한번에 스테이지에 올라감

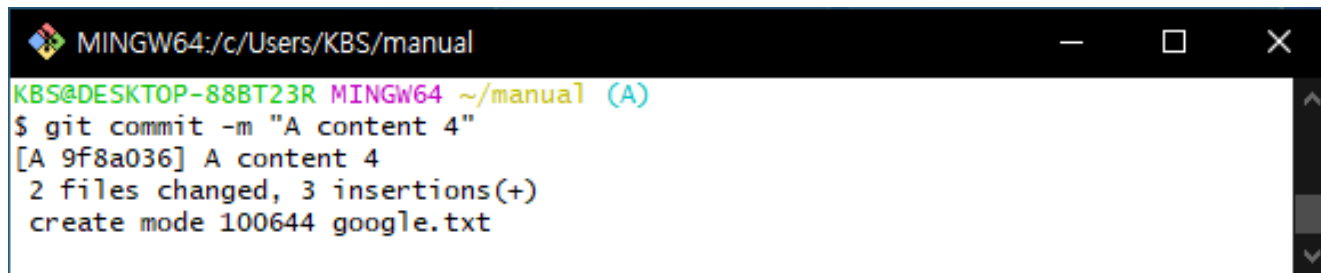


```
MINGW64:/c/Users/KBS/manual  
KBS@DESKTOP-88BT23R MINGW64 ~/manual (A)  
$ git status  
On branch A  
Changes to be committed:  
  (use "git restore --staged <file>..." to unstage)  
    new file:   hello.txt  
    modified:   work.txt
```


브랜치 변경 후 작업하기

❖ 커밋하기

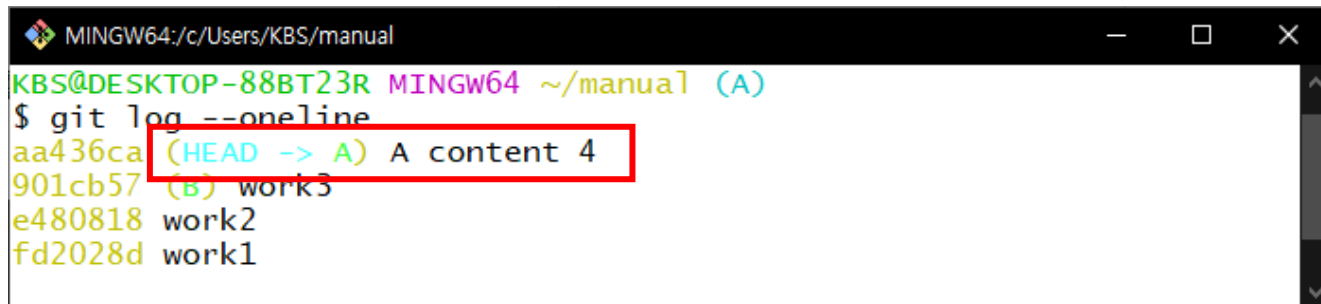
```
$ git commit -m "A content 4"
```



```
MINGW64:/c/Users/KBS/manual
KBS@DESKTOP-88BT23R MINGW64 ~/manual (A)
$ git commit -m "A content 4"
[A 9f8a036] A content 4
2 files changed, 3 insertions(+)
create mode 100644 google.txt
```

❖ 커밋 로그 간략하게 확인하기

```
$ git log --oneline
```



```
MINGW64:/c/Users/KBS/manual
KBS@DESKTOP-88BT23R MINGW64 ~/manual (A)
$ git log --oneline
aa436ca (HEAD -> A) A content 4
901cb57 (B) work3
e480818 work2
fd2028d work1
```



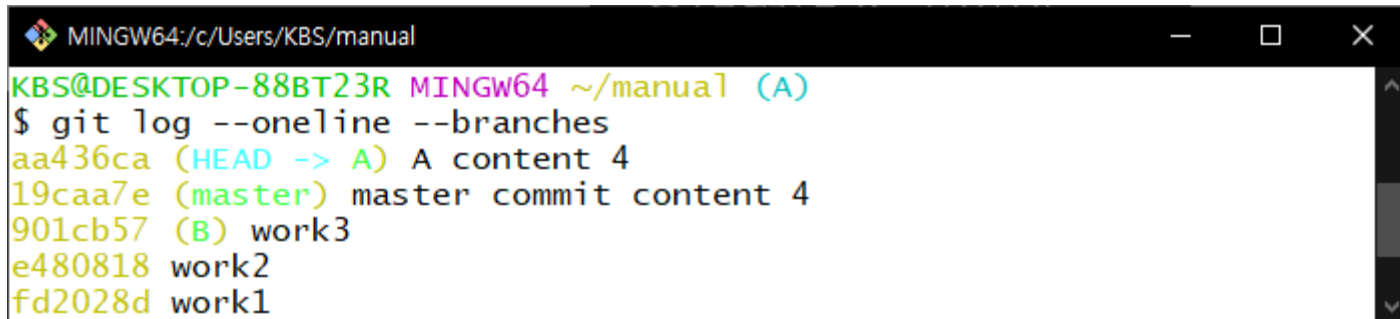
04

브랜치 정보 확인하기

브랜치 정보 확인하기

❖ 브랜치마다 커밋 로그 확인하기

```
$ git log --oneline --branches
```



The screenshot shows a terminal window titled 'MINGW64:/c/Users/KBS/manual'. The prompt is 'KBS@DESKTOP-88BT23R MINGW64 ~/manual (A)'. The command '\$ git log --oneline --branches' has been executed, resulting in the following output:

```
aa436ca (HEAD -> A) A content 4
19caa7e (master) master commit content 4
901cb57 (B) work3
e480818 work2
fd2028d work1
```

- A 브랜치 : A content 4
- master 브랜치 : master commit content 4
- B 브랜치 : work3

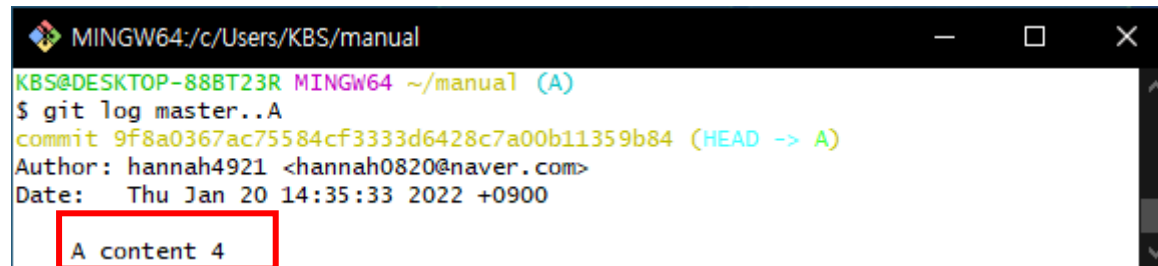
브랜치 정보 확인하기

❖ 브랜치 사이의 차이점 확인하기

```
$ git log 기준_브랜치..비교_브랜치
```

- 예) master 브랜치와 A 브랜치를 비교

```
$ git log master..A
```

A screenshot of a Windows terminal window titled 'MINGW64:/c/Users/KBS/manual'. The prompt is 'KBS@DESKTOP-88BT23R MINGW64 ~/manual (A)'. The command '\$ git log master..A' has been executed. The output shows a single commit: 'commit 9f8a0367ac75584cf3333d6428c7a00b11359b84 (HEAD -> A)' by 'Author: hannah4921 <hannah0820@naver.com>' on 'Date: Thu Jan 20 14:35:33 2022 +0900'. The commit message 'A content 4' is displayed and is highlighted with a red rectangular box.

```
MINGW64:/c/Users/KBS/manual
KBS@DESKTOP-88BT23R MINGW64 ~/manual (A)
$ git log master..A
commit 9f8a0367ac75584cf3333d6428c7a00b11359b84 (HEAD -> A)
Author: hannah4921 <hannah0820@naver.com>
Date: Thu Jan 20 14:35:33 2022 +0900
A content 4
```

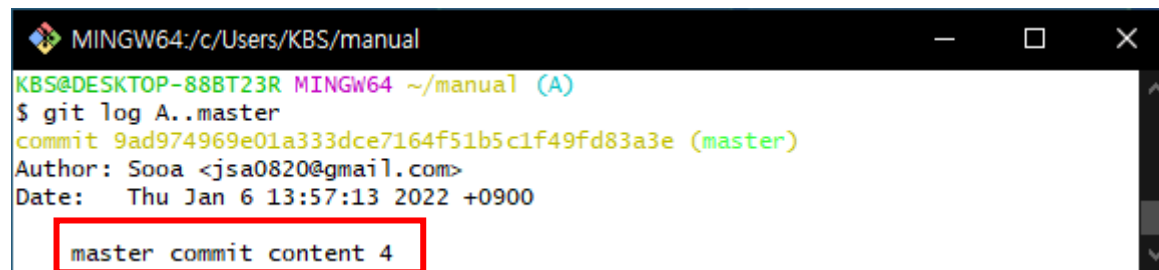
- master 브랜치에는 없고, A 브랜치에는 있는 커밋을 보여줌

브랜치 정보 확인하기

❖ 브랜치 사이의 차이점 확인하기

- 예) master 브랜치와 A 브랜치를 비교

```
$ git log A..master
```

A screenshot of a Windows terminal window titled 'MINGW64:/c/Users/KBS/manual'. The prompt is 'KBS@DESKTOP-88BT23R MINGW64 ~/manual (A)'. The command '\$ git log A..master' has been executed. The output shows a single commit: 'commit 9ad974969e01a333dce7164f51b5c1f49fd83a3e (master)' by 'Author: Sooa <jsa0820@gmail.com>' on 'Date: Thu Jan 6 13:57:13 2022 +0900'. The commit message 'master commit content 4' is displayed and is highlighted with a red rectangular box.

```
MINGW64:/c/Users/KBS/manual
KBS@DESKTOP-88BT23R MINGW64 ~/manual (A)
$ git log A..master
commit 9ad974969e01a333dce7164f51b5c1f49fd83a3e (master)
Author: Sooa <jsa0820@gmail.com>
Date: Thu Jan 6 13:57:13 2022 +0900
master commit content 4
```

- A 브랜치에는 없고, master 브랜치에는 있는 커밋을 보여줌

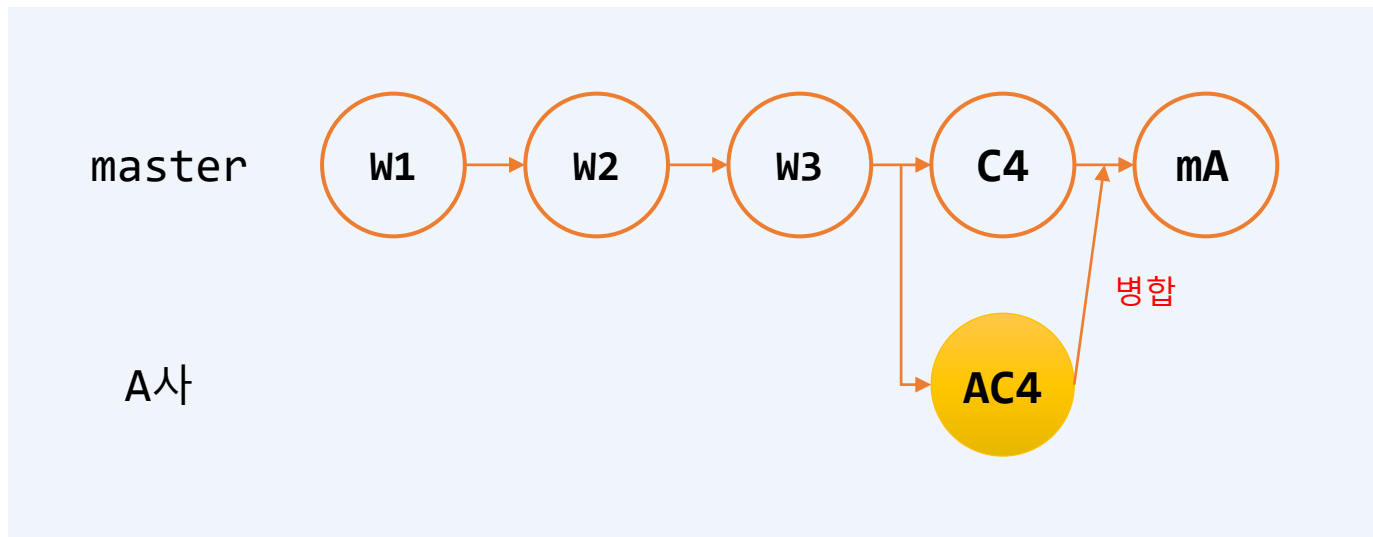


05

브랜치 병합하기

브랜치 병합하기

❖ 브랜치 병합



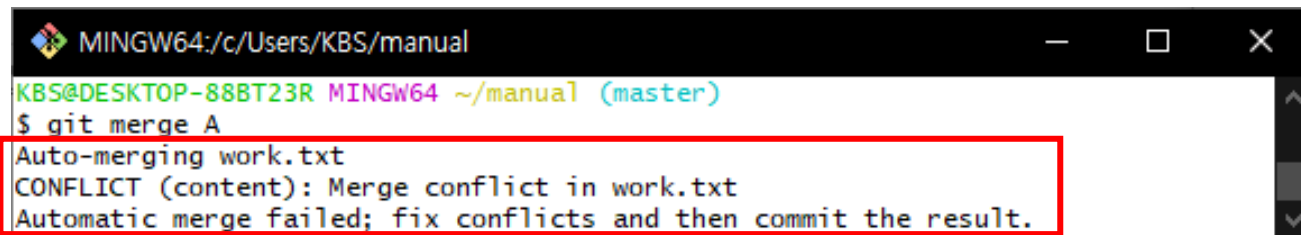
브랜치 병합하기

❖ master 브랜치로 이동하기

```
$ git switch master
```

❖ master 브랜치에 A 브랜치 병합하기

```
$ git merge A
```



```
MINGW64:/c/Users/KBS/manual
KBS@DESKTOP-88BT23R MINGW64 ~/manual (master)
$ git merge A
Auto-merging work.txt
CONFLICT (content): Merge conflict in work.txt
Automatic merge failed; fix conflicts and then commit the result.
```

The screenshot shows a Windows terminal window with the title 'MINGW64:/c/Users/KBS/manual'. The prompt is 'KBS@DESKTOP-88BT23R MINGW64 ~/manual (master)'. The user enters '\$ git merge A'. The output shows 'Auto-merging work.txt', followed by a conflict: 'CONFLICT (content): Merge conflict in work.txt'. The final message is 'Automatic merge failed; fix conflicts and then commit the result.', which is highlighted with a red box.

충돌 발생

브랜치 병합하기

❖ work.txt 파일 내용 확인

```
content 1  
content 2  
content 3  
<<<<<<< HEAD
```

```
content 4
```

현재 브랜치에서 수정한 내용

```
=====
```

```
A content 4
```

병합할 브랜치에서 수정한 내용

```
>>>>>>> A
```

브랜치 병합하기

❖ 파일 내용을 원하는 대로 수정하고 저장

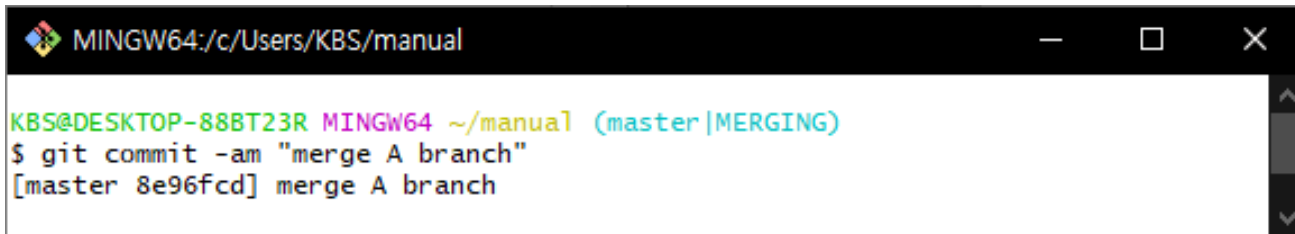
- <<<<<< HEAD, =====, >>>>>> A는 삭제

```
content 1  
content 2  
content 3  
  
content 4  
A content 4
```

브랜치 병합하기

❖ work.txt 파일 스테이징 및 커밋하기(mA)

```
$ git commit -am "merge A branch"
```

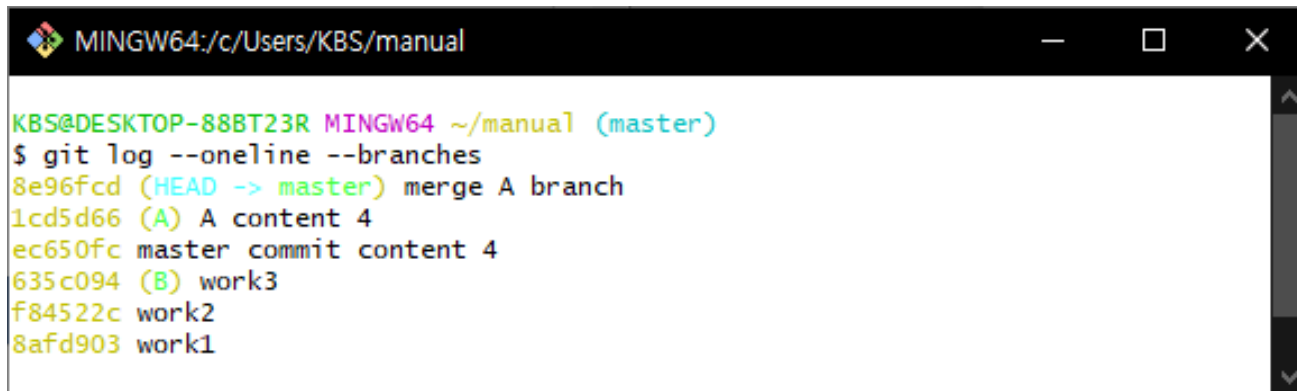
A screenshot of a Windows terminal window titled "MINGW64:/c/Users/KBS/manual". The terminal shows the user "KBS@DESKTOP-88BT23R" in a "MINGW64" environment at the directory "~/manual". The prompt is "(master |MERGING)". The user enters the command "\$ git commit -am 'merge A branch'", and the terminal responds with "[master 8e96fcd] merge A branch".

```
MINGW64:/c/Users/KBS/manual  
KBS@DESKTOP-88BT23R MINGW64 ~/manual (master |MERGING)  
$ git commit -am "merge A branch"  
[master 8e96fcd] merge A branch
```

브랜치 병합하기

❖ 로그 확인하기

```
$ git log --oneline --branches
```

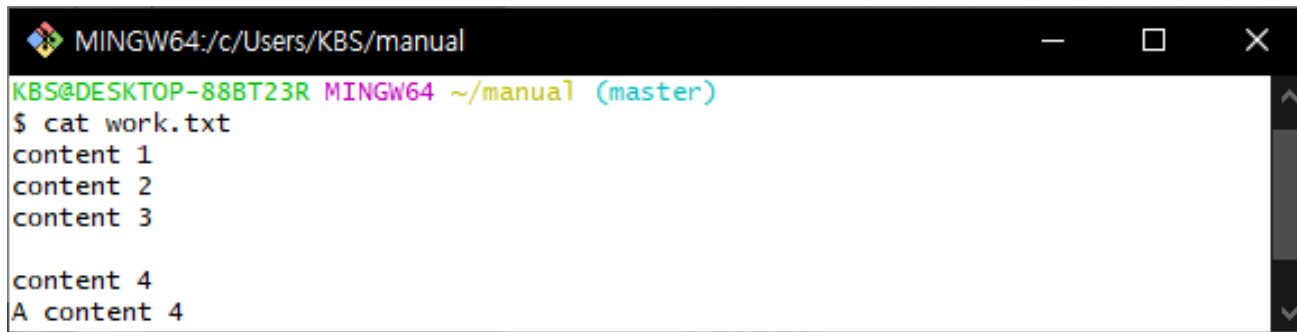
A screenshot of a Windows terminal window titled 'MINGW64:/c/Users/KBS/manual'. The terminal shows the output of the command 'git log --oneline --branches'. The output lists several commits with their hashes and branch names in parentheses. The commits are: '8e96fcd (HEAD -> master) merge A branch', '1cd5d66 (A) A content 4', 'ec650fc master commit content 4', '635c094 (B) work3', 'f84522c work2', and '8afd903 work1'. The terminal has a black background and a scrollbar on the right side.

```
MINGW64:/c/Users/KBS/manual  
KBS@DESKTOP-88BT23R MINGW64 ~/manual (master)  
$ git log --oneline --branches  
8e96fcd (HEAD -> master) merge A branch  
1cd5d66 (A) A content 4  
ec650fc master commit content 4  
635c094 (B) work3  
f84522c work2  
8afd903 work1
```

브랜치 병합하기

❖ work.txt 파일 확인하기

```
$ cat work.txt
```



```
MINGW64:/c/Users/KBS/manual
KBS@DESKTOP-88BT23R MINGW64 ~/manual (master)
$ cat work.txt
content 1
content 2
content 3

content 4
A content 4
```



06

정리하기

정리하기

<code>git branch newBranch</code>	새로운 브랜치 newBranch 생성
<code>git switch newBranch</code>	기존 브랜치에서 newBranch로 변경
<code>git log --oneline</code>	한 줄씩 커밋 로그 출력
<code>git add .</code>	수정한 전체 파일을 스테이징
<code>git log --oneline --branches</code>	브랜치 별 커밋 로그 출력
<code>git merge newBranch</code>	newBranch를 master 브랜치에 병합

THANK 😊 YOU