







- 1. 깃 설정과 저장소 생성
- 2. 커밋 과정
- 3. 이전 버전으로 이동 후 다시 최신 버전으로 이동



- 1. 깃 초기 설정하고 저장소를 생성할 수 있다.
- 2. 파일을 만들어 버전관리를 위한 과정을 수행할 수 있다.
- 3. 버전관리 로그 이력을 확인할 수 있다.
- 4. 이전 버전으로 이동하고 다시 최신 버전으로 이동할 수 있다.











	명령	설명
	\$ git configglobal user.name hskang	사용자 이름 설정
	\$ git configglobal user.email hskang@gmail.com	전자메일 설정 설정
1	<pre>\$ git configglobal core.autocrlf true \$ git configglobal core.safecrlf false</pre>	자동 줄바꿈 설정 안전 줄바꿈 설정
	<pre>\$ git configglobal core.editor 'codewait' \$ git configglobal init.defaultBranch main</pre>	기본 편집기 설정 기본 브랜치이름 설정
	\$ git init basic	저장소 생성
2	\$ cd basic \$ ls -al	폴더 이동 파일 확인







이전 이전 버전		이전 버전		최신 버전
HEAD~~ HEAD~2		HEAD~ HEAD~1		HEAD
HEAD^^ HEAD^~ HEAD~^		HEAD^ HEAD^1		•
main		main		main
2e11f34		2fcaf34		2341f34
Α		В		С
hello.txt		hello.txt		hello.txt
aaa	-	aaa bbb	—	aaa bbb ccc
\$ git checkout HEAD~2		\$ git checkout HEAD~		A
				\$ git checkout main \$ git checkout -





		작업디렉토리	스테이징 영역	깃 저장소
9.	\$ echo aaa > hello.txt	aaa		
① 파일 수정	<pre>\$ cat hello.txt \$ git status [long]</pre>			
10	\$ git status -s	?? hello.txt		
2	\$ git add hello.txt	aaa	aaa	
파일	\$ git status			
추가	\$ git status [short -s]	A hello.txt		
	\$ git commit -m A	aaa	aaa	aaa
3	\$ git status			
첫 번째 커밋	<pre>\$ git log \$ git logoneline \$ git log [patch -p]</pre>			



1 커밋과로그이력

🌣 총 2회의 커밋 이력과 수정

		작업디렉토리	스테이징 영역	깃 저장소
4	<pre>\$ echo bbb >> hello.txt</pre>	aaa bbb		
파일 수정	<pre>\$ cat hello.txt \$ git status [long]</pre>			
	\$ git status -s	M hello.txt		
	\$ git commit -am B	aaa bbb	aaa bbb	aaa bbb
⑤ ⊏ ₩₩	\$ git status			
두 번째 커밋	<pre>\$ git log \$ git logonline \$ git log [patch -p]</pre>			
⑥ 다시 파일 수정	<pre>\$ echo ccc >> hello.txt</pre>	aaa bbb ccc	aaa bbb	aaa bbb
	<pre>\$ cat hello.txt \$ git status \$ git status -s</pre>	M hello.txt		7





1 커밋과로그이력

🍅 총 3회의 커밋 이력과 수정, 추가, 수정

		작업디렉토리	스테이징 영역	깃 저장소
9-	\$ git commit —am C	aaa bbb ccc	aaa bbb ccc	aaa bbb ccc
⑦ 세 번째 커밋	<pre>\$ git status \$ git log \$ git logonline \$ git log [patch -p] \$ git logreverseoneline \$ git log -2 \$ git log -2oneline \$ git loggraphoneline \$ git show \$ git showoneline \$ git show HEAD \$ git show HEAD~2</pre>			





		작업디렉토리	스테이징 영역	깃 저장소
® 수정	\$ echo ddd >> hello.txt	aaa bbb ccc ddd	aaa bbb ccc	aaa bbb ccc
10	<pre>\$ git status \$ git status -s</pre>	M hello.txt		
⑨ 추가	\$ git add hello.txt	aaa bbb ccc ddd	aaa bbb ccc ddd	aaa bbb ccc
	<pre>\$ git status \$ git status -s</pre>	M hello.txt		
⑩ 수정	<pre>\$ echo eee >> hello.txt</pre>	aaa bbb ccc ddd eee	aaa bbb ccc ddd	aaa bbb ccc
	<pre>\$ git status \$ git status -s</pre>	M <mark>M</mark> hello.txt		9







2 과거이전버전으로이동



☑ 3번의 커밋이 수행된 상태와 커밋 로그 이력

작업 폴더	스테이지 영역	깃 저장소
(hello.txt)	(hello.txt)	(hello.txt)
aaa bbb ccc ddd eee	aaa bbb ccc ddd	aaa bbb ccc

С	HEAD
aaa bbb ccc	07af7f6
•	
В	HEAD~ HEAD^ HEAD~1 HEAD^1
aaa bbb	3a2d414
•	
Α	HEAD~~ HEAD^^ HEAD~^ HEAD~2
aaa	dd3f28a





과거이전버전으로이동



- ★ \$ git checkout [이전커밋]
 - 상태가 clean해야 함
 - 3 영역이 동일한 상태
- ❤️ 현재 상태를 clean한 상태로 하는 방법
 - 임시 저장인 git stash
 - 작업 디렉토리와 스테이징 영역의 저장하고 3 영역이 동일한 상태가 되도록 후에 다시 자세히 학습 예정

	작업디렉토리	스테이징 영역	깃 저장소
	aaa bbb	aaa bbb	aaa bbb
현재 상황	ccc ddd	ccc ddd	CCC
\$ git checkout HEAD~	eee 오류 발생		
\$ git stash	aaa bbb	aaa bbb	aaa bbb
\$ cat hello.txt	ССС	ссс	ссс





2 과거이전버전으로이동

🐡 과거 버전 이동 준비



☑ 스테이징 영역은 커밋과 같게

		작업디렉토리	스테이징 영역	깃 저장소
	현재 상황	aaa bbb ccc ddd eee	aaa bbb ccc ddd	aaa bbb ccc
준비	\$ git checkout HEAD~	오류 발생		
	\$ git stash 위는 다음 2개의 명령으로도 같은 기능 수행 \$ git restorestaged hello.txt \$ git restore hello.txt	aaa bbb ccc	aaa bbb ccc	aaa bbb ccc
	<pre>\$ cat hello.txt</pre>			



2 과거이전버전으로이동 <u></u>



₩ 바로 이전으로 이동 후 내용 살피고 다시 원복

	작업디렉토리	스테이징 영역	깃 저장소
\$ git checkout HEAD~	aaa bbb	aaa bbb	aaa bbb
\$ git status			
\$ cat hello.txt			
\$ git log \$ git logoneline \$ git loggraph \$ git logalloneline			
\$ git checkout main	aaa bbb ccc	aaa bbb ccc	aaa bbb ccc
\$ cat hello.txt	000		
Α	В		С
hello.txt	hello.txt	h	ello.txt
naa	aaa bbb	aaa bbb ccc	
\$ git checkout HEAD~2	\$ git checkout HEAD~	\$ git o	
	\$ git status \$ cat hello.txt \$ git log \$ git logoneline \$ git loggraph \$ git logalloneline \$ git checkout main \$ cat hello.txt A hello.txt	\$ git checkout HEAD~ \$ git status \$ cat hello.txt \$ git log \$ git logoneline \$ git loggraph \$ git logalloneline \$ git checkout main \$ bbb ccc \$ cat hello.txt A B hello.txt aaa bbb	\$ git checkout HEAD~ \$ git status \$ cat hello.txt \$ git log \$ git logoneline \$ git loggraph \$ git logalloneline \$ git checkout main \$ bbb \$ ccc \$ cat hello.txt A B hello.txt A B hello.txt aaa bbb ccc





2 과거이전버전으로이동



설명	git checkout	git switch
이전 커밋으로 이동	\$ git checkout [이전커밋]	\$ git switch -d [이전커밋]
다른 브랜치로 이동	\$ git checkout [branch]	\$ git switch [branch]
새로운 브랜치를 생성하고 이동	\$ git checkout -b [newBranch]	\$ git switch -c [newBranch]





Summary

>> 저장소 생성

\$ git init basic

>> 깃 저장소에 저장

• \$ git add, commit

>> 모두 커밋 이력 보기

- \$ git log
 - --oneline --graph --all -n

>>> 특정한 커밋 이력 보기

- \$ git show [HEAD]
 - --oneline





Summary

- >> 바로 이전 버전으로 가기
 - \$ git checkout HEAD~
- >> 다시 최신 버전으로 돌아오기
 - \$ git checkout main
- >> 다시 checkout 이전으로 돌아오기
 - \$ git checkout -