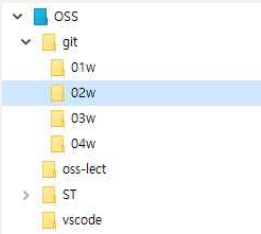


2주차 실습 시트

강 환수 교수

프로젝트	git basic: 저장소 생성과 커밋 이력 관리 [교재 2, 3, 4장]	
모듈1	<input type="checkbox"/> 리눅스와 깃의 기본 명령어 <input type="checkbox"/> 깃에서 전체 필요한 설정과 지역 저장소 설정과 확인	
모듈2	<input type="checkbox"/> 지역저장소 basic 생성 후 환경 설정과 해제, 저장소 삭제	
모듈3	<input type="checkbox"/> 지역저장소 mid를 생성 후, 파일 hello를 생성 후 1회 커밋, 이력 정보 확인	
모듈4	<input type="checkbox"/> 원격저장소 mid에 파일 hello에 2회 연속 커밋, add와 commit을 나누어 상태를 파악	
준비	<input type="checkbox"/> 적당한 드라이브에 다음 폴더를 생성 후 git/02w에서 git bash 실행 후 시작 <ul style="list-style-type: none"> ○ 현재의 폴더가 항상 명시 ○ \$: prompt, 명령을 기다릴 준비가 되어 있다는 의미 <div>  </div> <pre>PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/02w \$</pre>	
모듈1 실습 (15min)	<input type="checkbox"/> 리눅스와 깃의 기본 명령어 <ul style="list-style-type: none"> ○ 폴더 확인, ~ (홈 폴더, C:/users/[사용자]) ○ 깃의 전체 설정 파일 내용 확인, 버전과 도움말 <input type="checkbox"/> 깃에서 전체 필요한 설정과 확인 <ul style="list-style-type: none"> ○ line feed / carriage return 자동 변환 ○ 이름과 메일 주소 설정, 편집기 설정 	
	<pre>\$ pwd \$ ~ \$ cat ~/.gitconfig \$ cat ~/.gitconfig [filter "lfs"] clean = git-lfs clean -- %f smudge = git-lfs smudge -- %f process = git-lfs filter-process required = true [user] name = hskang email = ai7dnn@gmail.com [credential] helper = manager-core [difftool "sourcetree"] cmd = " \"\$LOCAL\" \"\$REMOTE\" " [mergetool "sourcetree"] cmd = "" " trustExitCode = true [core] longpaths = true autocrlf = true editor = code --wait safecrlf = false [init] defaultBranch = main [gui] recentrepo = C:/[git tutorial]/git-diff \$ clear</pre>	전체 설정 파일 위치 C:/user/[사용자] 전체 설정 위치 C:/user/[사용자] 전체 설정 파일 .gitconfig
	<pre>\$ git \$ git help 또는 --help</pre>	깃 전체 도움말 - 간단 옵션 -- 긴 옵션

	<pre>\$ git help -a 또는 --all</pre> <pre>\$ git help commit (웹브라우저에 도움말 페이지 표시)</pre> <pre>\$ git commit -h</pre> <pre>\$ git version 또는 --version</pre> <pre>\$ git version; git help</pre>	<p>화면이 보이는 것이 많은 경우, vi 스타일로 표시되고, : 이후에 [ctrl + f]를 눌러 다음 페이지가 더 표시되게 하고 [end] 보일 때 q(uit)를 누르고 종료, 중간에 표시를 종료하고 싶으면 바로 q</p> <p>commitd 대한 도움말 명령어; 명령어 = 한 줄에 여러 명령</p>
	<pre>\$ git config --global user.name ai7dnn</pre> <pre>\$ git config --global user.email ai7dnn@gmail.com</pre> <pre># 맥(lf)과 윈도우(crlf) 간의 자동 변환</pre> <pre>\$ git config --global core.autocrlf true</pre> <pre># 뉴라인 경고 발생 없애기(옵션)</pre> <pre>\$ git config --global core.safecrlf false</pre> <pre>\$ git config --global core.editor notepad</pre> <pre>\$ git config --global core.editor "code --wait"</pre>	<p>사용자 환경 설정</p> <p>[교재 p41] 윈도우와 맥의 캐리지 리턴과 라인 피드 변환 (윈도우와 맥과 차이)</p> <p>편집기 환경 설정 환경 설정 확인</p> <p>환경 설정 하나 하나 확인 방법</p>
	<pre># 기본 브랜치 지정, main으로</pre> <pre>\$ git config --global init.defaultBranch main</pre> <pre>\$ git config --global --list</pre> <pre>\$ git config init.defaultBranch</pre> <pre>\$ git config --get init.defaultBranch</pre> <pre>\$ git init test</pre> <pre>\$ cd test</pre> <pre># (main)으로 지정된 것을 확인</pre>	<p>기본 설치 시 기본 브랜치 이름이 master로 되어 있다면 앞으로 생성될 저장소의 기본 브랜치 이름을 main으로 수정하기 위한 설정</p> <p>이미 만든 것이 바뀌는 것이 아니라 새로 만들어지는 저장소의 브랜치 이름을 main으로 생성됨</p>
	<p>화면 대비 결과가 많은 경우: vim 편집기로 표시 : + i => 수정, : + w => 저장, : + q => 종료 [end] 표시 나오면 바로 q 입력하면 종료</p> <ul style="list-style-type: none"> • vi 명령어(커밋 메시지 등을 작성할 때 사용) <ul style="list-style-type: none"> • i: 입력 상태로 전환 • w: 저장 • q: 종료 • wq: 저장 후 종료 • q!: 저장하지 않고 종료 • esc: 입력 상태에서 명령 모드로 전환 	<p>Vim(Vi IMproved)은 브람 물리너(Bram Moolenaar)가 1991년에 만든 vi 호환 텍스트 편집기</p> <p>[교재 p40] 편집 시에는 esc 하면 :이 나타남 : + [ctrl] + F => 다음 화면에 표시</p>
	<pre>\$ git config --global -e</pre>	전역 설정 파일 .gitconfig 편집
모듈2 실습 (15min)	□ 지역저장소 basic 생성 후 환경 설정과 해제, 저장소 삭제	
	<pre>\$ git init basic</pre>	저장소 basic 생성
	<pre>\$ cd basic</pre> <pre>\$ ls -al</pre>	저장소 이동 후 하부 파일과 숨김 폴더(.git) 확인
	<pre>\$ git config user.name hskang</pre> <pre>\$ git config user.email hs kang@dongyang.ac.kr</pre> <pre>\$ git config -l 또는 \$ git config --list</pre>	<p>지역 환경 설정 지역 환경 설정 보기 지역 환경 설정 파일(.git/config)</p>

	\$ cat .git/config	직접 보기
	\$ git config user.email \$ git config user.name hong \$ git config user.name \$ git config --unset user.name \$ git config user.name \$ git config --get user.name \$ git config -help 또는 --help	지역 환경 설정 user.name 보기 지역 환경 설정 user.name 재 설정 지역 환경 설정 user.name 보기 [교재 p58] 지역 환경 설정 user.name 해제 지역 환경 설정 user.name 보기 깃 설정 도움말
	\$ git config -e	지역 설정 파일 .git/config 편집
	\$ rm -rf .git \$ cd .. \$ rm -rf basic	저장소 하부 폴더 .git 삭제: 저장소 기능 상실 저장소 폴더 삭제
모듈3 실습 (15min)	□ 지역저장소 mid를 생성 후, 파일 hello를 생성 후 1회 커밋, 이력 정보 확인	
	\$ git init mid \$ cd mid \$ ls -al	저장소 하부 폴더 .git: 실제 깃 저장소, 커밋과 브랜치 변경 내용을 저장, 이 폴더가 없으면 저장소 기능 상실
	PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/02w/my (main) \$ git config core.autocrlf true	다음 4개 설정 반드시 확인 core.autocrlf true core.safecrlf false user.name hskang user.email hskang@dongyang.ac.kr
	PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/02w/my (main) \$ git config core.safecrlf false	없으면 설정 필요
	PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/02w/my (main) \$ git config user.mail	
	PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/02w/my (main) \$ git config user.email ai7dnn@gmail.com PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/02w/my (main) \$ git config user.name ai7dnn	
	\$ echo 'A' > hello \$ cat hello \$ git status \$ git status -s 또는 --short \$ git status On branch main No commits yet Untracked files: (use "git add <file>..." to include in what will be committed) hello nothing added to commit but untracked files present (use "git add" to track) \$ git status -s ?? hello	> 파일 수정 기존 파일, 이전에 있던 내용을 모두 제거하고 삽입 기존 없던 파일, 새로운 파일 생성해 삽입 [교재 78, 97] 파일 생성(untracked file): 작업공간에 새로운 파일이 생성 상태 보기, 짧게 보기
	\$ git add hello	스테이지(stage)에 저장(tracked file)

<pre>\$ git status On branch main No commits yet Changes to be committed: (use "git rm --cached <file>..." to unstage) new file: hello \$ git status -s A hello</pre>	<p>처음으로 add되면 A hello</p> <p>• ?? => A□ - Untracked에서 SA로 add된 상태의 변화</p>
<pre>\$ git commit -m 'create hello' \$ git commit -m 'create hello' [main (root-commit) 7ac481f] create hello 1 file changed, 1 insertion(+) create mode 100644 hello \$ git status On branch main nothing to commit, working tree clean \$ git status -s</pre>	<p>깃 저장소(git repository) 저장, commit: '~에 적어두다'를 의미</p> <p>WD와 SA가 동일하면 출력 없음</p>
<pre>\$ git log \$ git log commit 7ac481fc6df3e78e3f6de963086f28383762ca98 (HEAD -> main) Author: hskang <ai7dnn@gmail.com> Date: Tue Sep 6 09:58:14 2022 +0900 create hello</pre>	<p>최근의 커밋 이력부터 이전의 모든 커밋을 보여주나, 여기서는 커밋이 하나이므로 하나만 보여줌</p> <p>깃 커밋 이력 정보 commit ID(40개의 16진수), (HEAD -> 브랜치명)</p> <p>HEAD: 브랜치의 마지막 커밋을 가리키는 포인터</p>
<pre>PC@DESKTOP-482NOAB MINGW64 /c/oss/git/02w/mid (main) \$ git log commit 65117423c9f95643343ccd6acee9dde041141a24 (HEAD -> main) Author: hskang <ai7dnn@gmail.com> Date: Sun Aug 28 13:12:24 2022 +0900 create hello</pre>	
<pre>\$ git log --oneline \$ git log --oneline 7ac481f (HEAD -> main) create hello</pre>	<p>커밋 이력 한 줄 확인</p>
<pre>PC@DESKTOP-482NOAB MINGW64 /c/oss/git/02w/mid (main) \$ git log --oneline 0cc6c16 (HEAD -> main) create hello</pre>	
<pre>\$ git show head # HEAD 포인터가 가리키는 커밋 정보를 확인 commit 7ac481fc6df3e78e3f6de963086f28383762ca98 (HEAD -> main) Author: hskang <ai7dnn@gmail.com> Date: Tue Sep 6 09:58:14 2022 +0900 create hello diff --git a/hello b/hello new file mode 100644 index 0000000..f70f10e --- /dev/null +++ b/hello @@ -0,0 +1 @@ +A</pre>	<p>로그에 대한 자세한 정보 보기, 다음도 가능</p> <p>git show : 현재 브랜치의 가장 최근 커밋 정보를 확인</p> <p>git show 커밋해시값 : 특정 커밋 정보를 확인</p> <p>\$ git show : 가장 최근의 커밋의 파일에 대한 자세한 정보</p>
<pre>\$ git config --global alias.log1 'log --oneline' \$ git config alias.log1 \$ git log1 \$ git config --global alias.logg1 'log --graph --oneline' \$ git logg1 \$ git config --global alias.sh show</pre>	<p>깃 명령 별칭 저장 저장된 별칭 조회 별칭 명령 실행</p>

	\$ git sh	<pre>PC@DESKTOP-482NOAB MINGW64 /c/oss/git/02w/mid (main) \$ git log --oneline 0cc6c16 (HEAD -> main) create hello PC@DESKTOP-482NOAB MINGW64 /c/oss/git/02w/mid (main) \$ git config --global alias.log1 'log --oneline' PC@DESKTOP-482NOAB MINGW64 /c/oss/git/02w/mid (main) \$ git log1 0cc6c16 (HEAD -> main) create hello</pre>
<div> <div>모듈4</div> <div>실습</div> <div>(20분)</div> </div>	<div> <div></div> <div>원격저장소 mid에 파일 hello에 2회 연속 커밋, add와 commit을 나누어 상태를 파악</div> </div>	
	<pre>\$ echo 'B' >> hello \$ cat hello A B \$ git sts M hello \$ git st On branch main 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 no changes added to commit (use "git add" and/or "git commit -a")</pre>	<p>기존 파일에 추가 >> 파일이 이미 있다면 마지막 줄에 추가</p> <p>첫 번째 열: SA의 상태 두 번째 열: WD의 상태</p> <ul style="list-style-type: none"> □□ => □M - 커밋한 이후에 파일 수정 시 상태의 변화
	<pre>\$ git commit -am 'update to add B on hello' [main 848716f] update to add B on hello 1 file changed, 1 insertion(+) \$ git st On branch main nothing to commit, working tree clean \$ git sts</pre>	<p>옵션 -am: add 후 커밋 메시지(add + commit을 한 번에 수행)</p> <p>수정한 파일만 가능, 처음 만든 untracked 파일은 add한 후 commit</p>
	<pre>\$ git log \$ git log HEAD \$ git log1 또는 \$ git log --oneline \$ git log commit 848716f9244328b2560cbeb812750a35d327815b (HEAD -> main) Author: hskang <ai7dnn@gmail.com> Date: Wed Sep 7 08:34:57 2022 +0900 update to add B on hello commit 7ac481fc6df3e78e3f6de963086f28383762ca98 Author: hskang <ai7dnn@gmail.com> Date: Tue Sep 6 09:58:14 2022 +0900 create hello \$ git log1 848716f (HEAD -> main) update to add B on hello 7ac481f create hello \$ git show head commit 848716f9244328b2560cbeb812750a35d327815b (HEAD -> main) Author: hskang <ai7dnn@gmail.com> Date: Wed Sep 7 08:34:57 2022 +0900 update to add B on hello diff --git a/hello b/hello</pre>	<p>깃 커밋 이력 정보</p> <p>HEAD 하나 전 이력 HEAD^ == HEAD~ HEAD^ == HEAD~ == HEAD~1</p> <p>HEAD 두 개 전 이력 HEAD^^ == HEAD~~ == HEAD~2</p>

	<pre> index f70f10e..35d242b 100644 --- a/hello +++ b/hello @@ -1 +1,2 @@ A +B \$ git show head^ commit 7ac481fc6df3e78e3f6de963086f28383762ca98 Author: hskang <ai7dnn@gmail.com> Date: Tue Sep 6 09:58:14 2022 +0900 create hello diff --git a/hello b/hello new file mode 100644 index 0000000..f70f10e --- /dev/null +++ b/hello @@ -0,0 +1 @@ +A </pre>	
	<pre> \$ echo 'C' >> hello \$ cat hello </pre>	워킹디렉토리(작업공간, working tree)의 파일을 수정
	<pre> \$ git status </pre>	파일 상태: modified
	<pre> PC@DESKTOP-482NOAB MINGW64 /c/oss/git/02w/mid (main) \$ git status On branch main 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 no changes added to commit (use "git add" and/or "git commit -a") </pre>	
	<pre> \$ git add hello \$ git status \$ git sts M hello </pre>	스테이지(stage: 커밋 준비가 저장된 공간)에 저장, 파란색 modified(커밋 준비가 된 것을 의미)
	<pre> PC@DESKTOP-482NOAB MINGW64 /c/oss/git/02w/mid (main) \$ git status On branch main Changes to be committed: (use "git restore --staged <file>..." to unstage) modified: hello </pre>	
	<pre> \$ git commit -m 'update to add C in hello' \$ git status </pre>	깃 저장소(git repository)에 저장, 메시지가 필요
	<pre> PC@DESKTOP-482NOAB MINGW64 /c/oss/git/02w/mid (main) \$ git commit -m 'update to add C in hello' [main 5e52b5] update to add C in hello 1 file changed, 1 insertion(+) PC@DESKTOP-482NOAB MINGW64 /c/oss/git/02w/mid (main) \$ git status On branch main nothing to commit, working tree clean </pre>	
	<pre> \$ git log \$ git log --online </pre>	

	<pre> PC@DESKTOP-482NOAB MINGW64 /c/oss/git/02w/mid (main) \$ git log commit 5e552b54e414eb840f04336dd11c94cd3e5c8c57 (HEAD -> main) Author: hskang <ai7dnn@gmail.com> Date: Sun Aug 28 13:37:22 2022 +0900 update to add c in hello commit f3fb2281ab0e20559d0d9d96df0da833dc55c788 Author: hskang <ai7dnn@gmail.com> Date: Sun Aug 28 13:21:06 2022 +0900 update to add B in hello commit 65117423c9f95643343ccd6acee9dde041141a24 Author: hskang <ai7dnn@gmail.com> Date: Sun Aug 28 13:12:24 2022 +0900 create hello PC@DESKTOP-482NOAB MINGW64 /c/oss/git/02w/mid (main) \$ git log --oneline 5e552b5 (HEAD -> main) update to add c in hello f3fb228 update to add B in hello 6511742 create hello </pre>	
	<pre> \$ git show head~2 commit 7ac481fc6df3e78e3f6de963086f28383762ca98 Author: hskang <ai7dnn@gmail.com> Date: Tue Sep 6 09:58:14 2022 +0900 create hello diff --git a/hello b/hello new file mode 100644 index 0000000..f70f10e --- /dev/null +++ b/hello @@ -0,0 +1 @@ +A </pre> <p>\$ git show head^2 fatal: ambiguous argument 'head^2': unknown revision or path not in the working tree. Use '--' to separate paths from revisions, like this: 'git <command> [<revision>...] -- [<file>...']'</p>	<p>\$ git show [커밋ID] 지정한 커밋ID에 대한 파일의 변화 등 자세한 정보를 보여줌</p> <p>HEAD 두 개 전 이력 HEAD^^ == HEAD~~ == HEAD~2</p> <p>다음은 사용 불가 HEAD^2</p>