

3주차 실습 시트

강 환수 교수

| 프로젝트 | commit & log, clone & pull [교재 4장 5장] | |
|----------------------|---|--|
| 모듈1 | <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, 별칭 명령어 설정 <input type="checkbox"/> 파일 hello.py 생성 및 실행, 삭제 <ul style="list-style-type: none"> ○ 추가되지 않은 파일은 탐색기에서 그대로 삭제 가능 | |
| 모듈2 | <input type="checkbox"/> 다시 파일 생성해 추가한 후 stage에서 삭제, 복구 후 커밋 | |
| 모듈3 | <input type="checkbox"/> 파일 basic.py 총 3회 커밋, 이력 정보 확인 | |
| 모듈4 | <input type="checkbox"/> 다양한 커밋 정보 확인, --patch --stat | |
| 모듈5 | <input type="checkbox"/> 깃허브 원격저장소 복제와 원격저장소 수정내용 가져오기(pull) | |
| 모듈6 | <input type="checkbox"/> 자신의 깃허브에 PAT(Personal Access Token) 생성, push와 pull | |
| 준비 | <input type="checkbox"/> 다음 폴더를 생성 후 git/03w에서 git bash 실행 후 시작 <input type="checkbox"/> vscode, github, source tree 준비 | |
| 모듈1 실습 (15min) | <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 <input type="checkbox"/> 파일 hello.py 생성 및 실행, 삭제 <ul style="list-style-type: none"> ○ 추가되지 않은 파일은 탐색기에서 그대로 삭제 | |
| | <pre>\$ git config --global user.name ai7dnn \$ git config --global user.email ai7dnn@gmail.com</pre> | 사용자 환경 설정 |
| | <pre>\$ git config --global core.autocrlf true \$ git config --global core.safecrlf false</pre> | [교재 p41] 윈도우와 맥의 캐리지 리턴과 라인 피드 변환 (윈도우와 맥과 차이) |
| | <pre>\$ git config --global core.editor "code --wait" \$ git config --global init.defaultbranch main</pre> | 편집기 환경 설정 환경 설정 확인 |
| | <pre>\$ git config --global alias.st status \$ git config --global alias.sts 'status -s'</pre> | |
| | <pre>\$ git config --global --list</pre> | |
| | <pre>\$ git init adv \$ cd adv</pre> | 저장소 adv 생성 |
| | <pre>\$ code hello.py</pre> | vscode로 파이썬 파일 편집 후 저장 |
| | <pre>\$ python hello.py</pre> | print(list('python')) (ctrl + s) 후 실행 |
| | <pre>\$ git config --global alias.st 'status' \$ git st</pre> | 별칭을 생성해 실행 |
| | <pre>\$ git st On branch main No commits yet Untracked files: (use "git add <file>..." to include in what will be committed) hello.py nothing added to commit but untracked files present (use "git add" to track)</pre> | 새로 생성하면 untracked file |

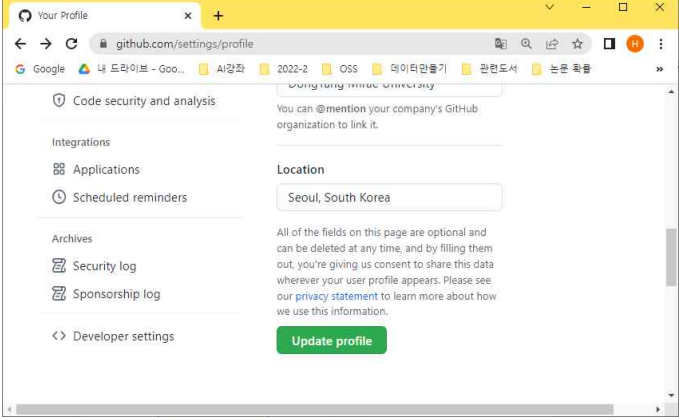
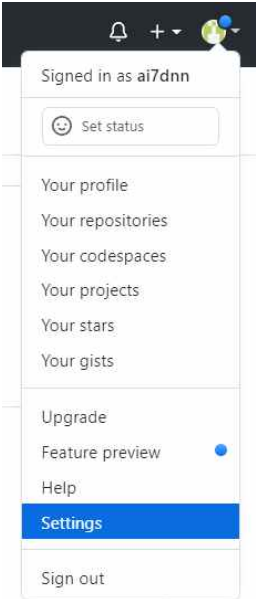
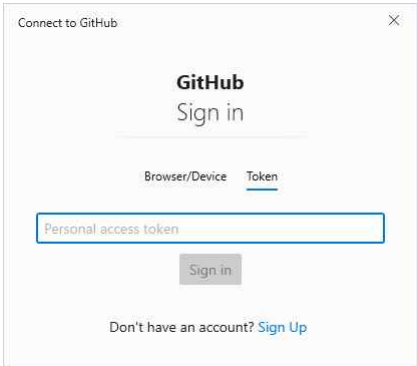
| | | |
|--|---|---|
| | <pre>\$ rm hello.py \$ ls \$ git st</pre> | untracked file은 그대로 삭제 가능 |
| | <pre>\$ git st On branch main No commits yet nothing to commit (create/copy files and use "git add" to track)</pre> | 저장소에 커밋할 것이 없음 |
| <div> <div>모듈2</div> <div>실습</div> <div>(20min)</div> </div> | □ 다시 파일 생성해 추가한 후 stage에서 삭제, 복구 후 커밋 | |
| | <pre>\$ code basic.py \$ python basic.py \$ git st</pre> | <p>저장소 basic 생성해 저장(ctrl + s) print(list(range(10)))</p> <p>깃 상태 보기</p> |
| | <pre>\$ git st On branch main No commits yet Untracked files: (use "git add <file>..." to include in what will be committed) basic.py nothing added to commit but untracked files present (use "git add" to track)</pre> | untracked file: basic.py |
| | <pre>\$ git add basic.py \$ git st</pre> | |
| | <pre>\$ git st On branch main No commits yet Changes to be committed: (use "git rm --cached <file>..." to unstage) new file: basic.py</pre> | 처음으로 스테이지에 들어온 파일: new file: basic.py |
| | <p>파일 수정 후 실행</p> <pre>\$ code basic.py # 또는 vscode 직접 수정 후 저장 \$ python basic.py \$ git st</pre> | <p>다음 코드 추가</p> <pre>print(list(range(1, 20, 3)))</pre> |
| | <pre>\$ git st On branch main No commits yet Changes to be committed: (use "git rm --cached <file>..." to unstage) new file: basic.py 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: basic.py</pre> | <p>하나의 파일이 2개로 나뉨</p> <p>new file: basic.py(stage 위치파일) print(list(range(10)))</p> <p>modified: basic.py(WD 위치파일) print(list(range(10))) print(list(range(1, 20, 3)))</p> |
| | <pre>PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git rm --cached basic.py error: the following file has staged content different from both the file and the HEAD: basic.py (use -f to force removal) PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git rm --cached -f basic.py rm 'basic.py' PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git st On branch main No commits yet Untracked files: (use "git add <file>..." to include in what will be committed) basic.py nothing added to commit but untracked files present (use "git add" to track)</pre> | <p>\$ git rm --cached fname 작업공간에는 남겨두고 스테이징 영역에서만 파일 삭제 파일 fname이 untracked 됨</p> <p>현재는 basic.py 가 WD와 stage가 다르므로 삭제가 불가하여, 무조건 삭제 옵션 -f 사용 삭제</p> <p>stage에서 제거하면 다시 untracked file이 되며, 내용은 위 modified file의 내용</p> <pre>print(list(range(10))) print(list(range(1, 20, 3)))</pre> |

| | | |
|--|---|--|
| | <pre>PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ cat basic.py print(list(range(10))) print(list(range(1, 20, 3)))</pre> | |
| | <pre>\$ git add basic.py \$ git st \$ git rm --cached basic.py \$ git st \$ git sts</pre> | <p>[교재 102, 103] stage에서 취소 stage -> untracked file</p> |
| | <pre>\$ git add basic.py PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git st On branch main No commits yet Changes to be committed: (use "git rm --cached <file>..." to unstage) new file: basic.py PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git rm --cached basic.py rm 'basic.py' PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git st On branch main No commits yet Untracked files: (use "git add <file>..." to include in what will be committed) basic.py nothing added to commit but untracked files present (use "git add" to track)</pre> | <p>stage area와 WD의 파일이 같으므로 git rm --cached basic.py 이 순조로이 가능</p> |
| | <pre>\$ git add basic.py \$ git commit -m 'create basic.py' \$ git log</pre> | |
| | <pre>\$ git log commit 8e28c32d4b1e48c81b0af8253ada5c314b1d5b60 (HEAD -> main) Author: hskang <ai7dnn@gmail.com> Date: Mon Aug 29 17:50:44 2022 +0900 create basic.py</pre> | |
| | <pre>PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git st On branch main nothing to commit, working tree clean PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git rm --cached basic.py rm 'basic.py' PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git st On branch main Changes to be committed: (use "git restore --staged <file>..." to unstage) deleted: basic.py Untracked files: (use "git add <file>..." to include in what will be committed) basic.py PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git sts D basic.py ?? basic.py</pre> | <p>커밋 이후 stage에서 삭제하면 다음 상태가 됨 D basic.py # SA엔는 삭제된 상태(D) ?? basic.py # WD에는 남아 untracked로 있 음</p> <p>파일이 삭제되었다는 내역이 Staging Area에 추가됨과 동시에 원본 파일이 디스크에만 남게되고 추적에서 제외되었기 때문에 untracked 됨</p> <p>https://dololak.tistory.com/310</p> |
| | <pre>PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git restore --staged basic.py PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git st On branch main nothing to commit, working tree clean PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ cat basic.py</pre> | <p>다시 SA를 이전 상태로 바꾸는 명령, 결과적으로 이전 파일로 다시 SA가 됨</p> |

| | | |
|----------------------|---|--|
| | <pre>print(list(range(10))) print(list(range(1, 20, 3)))</pre> | |
| 모듈3 실습 (15min) | □ 파일 basic.py 총 3회 커밋, 이력 정보 확인 | |
| | <pre>\$ # 편집 \$ git st</pre> | 파일 basic.py , 다음 추가 편집 저장 <pre>print([i for i in range(10)])</pre> |
| | <pre>\$ 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: basic.py no changes added to commit (use "git add" and/or "git commit -a")</pre> | |
| | <pre>\$ git commit -am 'add comp list to basic.py' \$ git st</pre> | 스테이지(stage)에 저장(tracked file)과 커밋을 한 번에 |
| | <pre>\$ git st On branch main nothing to commit, working tree clean</pre> | 깃 저장소(git repository) 저장, commit: '~에 적어두다'를 의미 |
| | <pre>\$ git config --global alias.logg1 'log --oneline --graph' \$ git logg1</pre> | 로그 확인 |
| | <pre>\$ git logg1 * a54df51 (HEAD -> main) add comp list to basic.py * 8e28c32 create basic.py</pre> | 깃 커밋 이력 정보 commit ID(7개의 16진수), (HEAD -> 브랜치명) 커밋메시지 |
| | <pre>\$ # 편집 후 저장 \$ python basic.py \$ git commit -am 'add comp dict to basic.py' \$ git log \$ git log basic.py \$ git logg1</pre> | 파일 basic.py 편집 저장 <pre>print({i:i*i for i in range(10)})</pre> 커밋 전체 로그 파일에 대한 커밋 전체 로그 커밋 전체 로그 한 줄씩 |
| 모듈4 실습 | <pre>\$ git logg1 * c038159 (HEAD -> main) add comp dict to basic.py * a54df51 add comp list to basic.py * 8e28c32 create basic.py</pre> | 3개의 커밋이 보임 |
| | □ 다양한 커밋 정보 확인 | |
| | <pre>\$ git log commit c03815919954ae0fcd8ec2abc8d8de75af922e3 (HEAD -> main) Author: hskang <ai7dnn@gmail.com> Date: Mon Aug 29 16:44:59 2022 +0900 add comp dict to basic.py</pre> | 커밋ID 저자 날짜 커밋메시지 \$ git log head^ 이전 커밋부터 모두 표시, 그러므로 2개만 표시 |
| | <pre>\$ git log --pretty=short commit c03815919954ae0fcd8ec2abc8d8de75af922e3 (HEAD -> main) Author: hskang <ai7dnn@gmail.com> add comp dict to basic.py</pre> | [교재 126] 날짜만 없음 |
| | <pre>\$ git show head \$ git show c038</pre> | [교재 127] head: 가장 최근 커밋 ID 4개 자리 이상 기술 |
| | | a/basic.py: 이전 파일 b/basic.py: 현재(커밋) 파일 이전 파일 줄 2에서, 3개 줄 이후(현재) 파일 줄 2에서, 4개 줄 <pre>@@ -2,3 +2,4 @@</pre> 추가된 줄 <pre>+print({i:i*i for i in range(10)})</pre> |

| | | |
|--|---|--|
| | <pre>\$ git show c038 commit c03815919954ae0fcda8ec2abc8d8de75af922e3 (HEAD -> main) Author: hskang <ai7dnn@gmail.com> Date: Mon Aug 29 16:44:59 2022 +0900 add comp dict to basic.py diff --git a/basic.py b/basic.py index 9a92761..7ce82bd 100644 --- a/basic.py +++ b/basic.py @@ -2,3 +2,4 @@ print(list(range(10))) print(list(range(1, 20, 3))) print([i for i in range(10)]) +print({i:i*i for i in range(10)})</pre> | |
| | <pre>\$ git log --oneline \$ git log --pretty=oneline</pre> | |
| | <pre>\$ git log --oneline c038159 (HEAD -> main) add comp dict to basic.py a54df51 add comp list to basic.py 8e28c32 create basic.py PC@DESKTOP-482NOAB MINGW64 /c/oss/git/03w/adv (main) \$ git log --pretty=oneline c03815919954ae0fcda8ec2abc8d8de75af922e3 (HEAD -> main) add comp dict to basic.py a54df51abd8723536016b302c079f0cdeb1606f add comp list to basic.py 8e28c32d4b1e48c81b0af8253ada5c314b1d5b60 create basic.py</pre> | |
| | <pre>\$ git log -p 또는 --patch</pre> | <p>[교재 127] 모든 로그에서 수정한 줄을 비교</p> |
| | <pre>\$ git log -p commit c03815919954ae0fcda8ec2abc8d8de75af922e3 (HEAD -> main) Author: hskang <ai7dnn@gmail.com> Date: Mon Aug 29 16:44:59 2022 +0900 add comp dict to basic.py diff --git a/basic.py b/basic.py index 9a92761..7ce82bd 100644 --- a/basic.py +++ b/basic.py @@ -2,3 +2,4 @@ print(list(range(10))) print(list(range(1, 20, 3))) print([i for i in range(10)]) +print({i:i*i for i in range(10)}) commit a54df51abd8723536016b302c079f0cdeb1606f Author: hskang <ai7dnn@gmail.com> Date: Mon Aug 29 16:31:16 2022 +0900 add comp list to basic.py diff --git a/basic.py b/basic.py index 3d44437..9a92761 100644 --- a/basic.py +++ b/basic.py @@ -1,2 +1,4 @@ print(list(range(10))) - print(list(range(1, 20, 3))) + print(list(range(1, 20, 3))) \ No newline at end of file + print(list(range(1, 20, 3))) + + print([i for i in range(10)])</pre> | <p>a/basic.py(이전'앞' 파일): 이전 커밋 b/basic.py(이후'뒤' 파일): 현재 커밋 @@ -2,3 +2,4 @@ -이전파일 2: 2줄(줄 번호)에서 다음 3개의 줄을 참조 +이후파일 2: 2줄(줄 번호)에서 다음 4개의 줄을 참조</p> |
| | <pre>PC@DESKTOP-482NOAB MINGW64 /c/OSS/git/03w/adv (main) \$ git log --stat commit 5c4d581dd76a1f91b022164d1b7d0a2818a018b7 (HEAD -> main) Author: ai7dnn <ai7dnn@gmail.com> Date: Sat Sep 17 11:41:54 2022 +0900 add comp dict to basic.py basic.py 3 ++- 1 file changed, 2 insertions(+), 1 deletion(-) commit d354b85c936d8364ced53a1c67d6b209ce33e321 Author: ai7dnn <ai7dnn@gmail.com> Date: Sat Sep 17 11:39:57 2022 +0900 add comp list to basic.py basic.py 4 +++- 1 file changed, 3 insertions(+), 1 deletion(-) commit 2546149b2676e7fe6caa62807a600a7be92a0a7c Author: ai7dnn <ai7dnn@gmail.com> Date: Sat Sep 17 11:07:20 2022 +0900</pre> | <p>간단한 통계 정보 표시 \$ git log --stat</p> |

| | | |
|-----------|--|---|
| | <pre>create basic.py basic.py 2 ++ 1 file changed, 2 insertions(+)</pre> | |
| 모듈5 실습 | <input type="checkbox"/> 깃허브 원격저장소 복제와 원격저장소 수정내용 가져오기(clone, pull) | |
| | <pre>\$ git clone 주소_URL # 동일 폴더 생성해 복제 \$ git clone 주소_URL . # 현재 폴더에 복제 \$ git clone 주소_URL dname # dname 폴더 생성해 복제 # 폴더 03w에서 작업 \$ cd .. \$ git clone https://github.com/ai7dnn/OSS-lect.git \$ git clone https://github.com/ai7dnn/OSS-lect.git Cloning into 'OSS-lect'... remote: Enumerating objects: 84, done. remote: Counting objects: 100% (84/84), done. remote: Compressing objects: 100% (82/82), done. remote: Total 84 (delta 50), reused 5 (delta 2), pack-reused 0 Receiving objects: 100% (84/84), 7.57 MiB 3.79 MiB/s, done. Resolving deltas: 100% (50/50), done.</pre> | <p>저장소와 동일 이름의 폴더 생성 현재 폴더에 저장소 생성 폴더 dname에 저장소 생성</p> <p>우리 수업 저장소 복제</p> |
| | <pre>\$ cd oss-lect # 폴더 이동 \$ git remote # 원격 저장소 별칭 이름 조회 \$ git remote -v # 원격 저장소 별칭 이름 세부 조회 \$ git pull # 수정된 내용 다시 가져오기 \$ git remote origin \$ git remote -v origin https://github.com/ai7dnn/OSS-lect (fetch) origin https://github.com/ai7dnn/OSS-lect (push) \$ git pull Already up to date. \$ git pull remote: Enumerating objects: 10, done. remote: Counting objects: 100% (10/10), done. remote: Compressing objects: 100% (8/8), done. remote: Total 9 (delta 1), reused 9 (delta 1), pack-reused 0 Unpacking objects: 100% (9/9), 738.45 KiB 1.99 MiB/s, done. From https://github.com/ai7dnn/OSS-lect f3d36c7..ab5064c main -> origin/main Updating f3d36c7..ab5064c Fast-forward .../02\354\243\274 add commit log.pdf" Bin 0 -> 407519 bytes ...4\230\201 \352\263\204\355\232\215\354\204\234.pdf" Bin 0 -> 72208 bytes ...4\227\205 \354\204\244\352\263\204\354\204\234.pdf" Bin 0 -> 165100 bytes ...4\235\230 \352\263\204\355\232\215\354\204\234.pdf" Bin 0 -> 93314 bytes ...4\235\230 \354\204\244\352\263\204\354\204\234.pdf" Bin 0 -> 112599 bytes 5 files changed, 0 insertions(+), 0 deletions(-)</pre> | <p>서버 원격 저장소 복제 후 변화한 것이 없다면</p> <p>\$ git pull Already up to date.</p> |
| 모듈6 실습 | <input type="checkbox"/> 자신의 깃허브에 PAT 생성 | |

| | | |
|--|--|--|
| |  <p>Settings / Developer settings</p> <p>GitHub Apps</p> <ul style="list-style-type: none"> OAuth Apps Personal access tokens <p>GitHub Apps</p> <p>Want to build something that integrates with and extends GitHub? Register a new GitHub App to get started developing on the GitHub API. You can also read more about building GitHub Apps in our developer documentation.</p> | <p>사용자 하부에서 settings</p>  <p>암호를 파일에 저장</p> |
| | | |
| <p>□ 저장소 repo-test 생성 후, 지역저장소와 연동하기</p> | | |
| | <pre>\$ git clone 주소</pre> <p>#### 파일 하나 생성</p> <pre>\$ git push</pre> <p>다음 화면 표시</p>  | <p>만일 현재 로그인되어 있다면 바로 Browser/Device 클릭</p> <p>Token 누르고 PAT 정확히 복사 후 sign in 클릭</p> |
| | <pre>\$ git push -u https://ghp_GeU8yVKNmKdIpZ787LHD6b6HARqF8h@github.com/lee7py/Python-Programming.git</pre> <pre>\$ git push -u https://{token}@github.com/{username}/{repo_name}.git</pre> | |