23-10-18

* Dos 시스템과 리눅스 시스템에서 파일 만들기
* Dos 시스템에서 파일 깃허브에 저장하는 방법

<리눅스 명령어>

* Sudo apt-get install 패키지명 //패키지명 설치
* Sudo apt-get install mariadb-server //mariadb-server 설치

1. apt-get => 설치목록
2. update => 목록
3. upgrade => 세부내용
4. remove => 삭제(패키지의 삭제)\_설정파일 남김

나중에 다시 설치할 때 자료남기기 위해서 사용

1. purge => 삭제(패키지의 삭제)\_설정파일안남김(같이제거)

나중에 다시 설치할 때 깔끔하게 제거되서 자료가 안남음

1. list => 조건목록

Q. 리눅스에서 사용할 때 삭제는 무엇을 사용할까요?

A. Purge 사용!

Remove를 사용하면 자료 기록이 남아서 손봐줘야 한다

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* **sudo dpkg -l | grep mariadb** //마리아DB 사용하는 모든 패키지를 보여준다

텍스트, 전자제품, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

텍스트, 전자제품, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

텍스트, 전자제품, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

Purge를 사용하여 mariadb 모든 패키지를 지웠다.

텍스트, 전자제품, 스크린샷, 디스플레이이(가) 표시된 사진

자동 생성된 설명

텍스트, 전자제품, 스크린샷, 디스플레이이(가) 표시된 사진

자동 생성된 설명

* ls -al //모든 파일 검색

텍스트, 전자제품, 스크린샷, 디스플레이이(가) 표시된 사진

자동 생성된 설명

* cd .ssh //.ssh로 이동
* cd .. //상위 단계로 이동

텍스트, 전자제품, 스크린샷, 디스플레이이(가) 표시된 사진

자동 생성된 설명

* tree //

텍스트, 스크린샷, 소프트웨어, 컴퓨터이(가) 표시된 사진

자동 생성된 설명

텍스트, 전자제품, 스크린샷, 디스플레이이(가) 표시된 사진

자동 생성된 설명

텍스트, 전자제품, 스크린샷, 디스플레이이(가) 표시된 사진

자동 생성된 설명

* mkdir test01 //test01이라는 디렉토리 만든다는 것
* rmdir test01 //test01이라는 디렉토리 삭제하는 것

텍스트, 전자제품, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* Touch test01.txt test02.txt test03.txt //여러 개 파일 한번에 만들 때 사용
* > test01.txt //파일 만드는데 명령어가 가장 짧다

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

텍스트, 전자제품, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* Cat > text01.txt //내용 입력하고 파일추가
* Cat >> text01.txt //기존에 있던 내용에 추가
* Cat //catch의 줄임말로, 파일 내용을 보여주는 명령어

Cat의 리다일액션이 가장편하다 추가적인 것이 필요할 때 vim으로 간다

텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

* Vim text01.txt //새 파일창 열어 편집할 수 있다

텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

스크린샷, 텍스트, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* Cp \*.\* test01 //모든파일명 디렉토리 파일 모두 복사한다

텍스트, 전자제품, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

패키지 설치, 권한 설정할 때 -> 최상위 명령어 sudo를 사용해야 함

파일이나 간단한 작업은 sudo 사용안해도 됨

Dos의 copy 대신 리눅스에서 cat 사용할 것이다

* Ps //프로세스
* Ps -e //내가 작업하는 모든 프로세스 나옴

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

Bash는 다시 생길 수 밖에 없다.

Kill 해도 다시 나온다

PID를 죽일 수 없어서 컴퓨터를 끄고 다시 켤 수밖에 없다.

* 오류 생기면 ps로 PID 강제로 죽이고 다시 작업하기

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

<명령 프롬프트 – Dos 명령어>

* md test01 //test01이라는 파일 만든다는 것
* rd test01 //test01이라는 파일 지운다는 것
* dir //디렉터리 보여주는 명령어

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* rd test01 /s //파일이 빈 파일이 아니라도 모두 지우겠다는 뜻이다

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* rd /? //rd에 대한 자세한 정보를 보여달라는 명령, 위에 rd /s 가 무엇인지 알 수 있음

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* copy 원본 대상
* copy con test01.txt //test01.txt 파일을 만든다 안에 내용 적을 수 있음

con : 콘솔 지금 층 에서는 키보드만 가능하다, 마우스 안됨

=>Dos 상에서 파일 만드는 첫번째 단계, 위에 입력하면 빈줄이 나옴 콘솔에서 입력한 것이 없으니까, 이후 ctrl + c 누르면 -> 종료가 된다

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* copy test01.txt test02.txt //test01.txt 파일을 test02.txt 파일로 복사해라

원본이 있으니까 원본을 복사하여 test01를 만들었다

텍스트, 디스플레이, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* copy test01.txt c:\ex\test01 //복사할거야 test01.txt 파일을 c:\ex\test01로(절대경로)
* copy test01.txt test01 //test01.txt 파일을 test01디렉토리로 복사할거야(상대경로)

텍스트, 디스플레이, 스크린샷, 컴퓨터이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* copy \*.txt test01 //ex디렉토리에 있는 모든 파일을 test01디렉토리로 복사함

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* cls //화면에 쓰고 있던 것 지워짐
* \* //다. 모두. All
* t\* //t로 시작하는 모든 파일명을 말한다
* \*t //t로 끝나는 모든 파일명을 말한다
* T? //T로 시작하는 두글자만 가능(T를 포함한다)
* ?T //T로 끝나는 두글자(T나옴 T앞에 아무것도 없을 수 있다)
* dir /? //dir의 도움말

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

<내용정리>

Java 설치후 => 환경변수 설정했다

* 우선권 부여 받아서 다른 프로그램 보다 상위 실행권을 획득하기 위해 A B 동시요청하는 경우가 생긴다
* 데드락(맞부딪힐 때) 컴퓨터 누구한테 먼저 작동할 것인가
* 우선권 부여할 것 -> 환경변수에서 획득하는 것

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

명령 프롬프트는 window에서 바뀐 내용들 바로 받지 못한다

Git 설치하고 안된 것은 바로 반영이 되지 않았기 때문 -> 껐다가 다시 켜니까 작동함

<git 과 git-hub>

Git 설치(64-bit Git for windows setup 으로 설치하기)

텍스트, 스크린샷, 소프트웨어, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

텍스트, 전자제품, 스크린샷, 디스플레이이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 컴퓨터, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* Git config --list //git이 설치되어 있는 최초의 환경설정 내용

텍스트, 전자제품, 스크린샷, 디스플레이이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

Git 이 있는 곳이 내가 저장하는 처음공간이다.

그래서 git-work안에 git이 있는 것보다 각 레파지토리(저장소)에 git이 있어야 한다. 그렇지 않으면 레파지토리를 기억도 하고 그 상위 폴더에서 변경된 사항도 다 저장하고 있는 것이다. 처음부터 다시 해야 한다면 git.폴더를 지우고 다시 생성하면 된다

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

Gitconfig를 검색하면 이런 파일이 나오고 intelliJ로 보면 이 파일에 대한 정보가 나온다

텍스트, 스크린샷, 소프트웨어, 번호이(가) 표시된 사진

자동 생성된 설명

Git-work안에 exm을 만들고 git에 대한 명령어 수행했을 때 git이 아래와 같이 만들어짐

텍스트, 스크린샷, 소프트웨어, 컴퓨터 아이콘이(가) 표시된 사진

자동 생성된 설명

window에서 git 설치, 명령프롬포트(cmd)에서 git 찾아냈다(숨김파일)

<cmd로 github에 파일 올리는 방법>

https://github.com/youjiyou82/TDN.git

git init // local git의 시작을 알림, github 레파지토리와 연결되는 곳

(다음에 들어올 때 할필요없다)

git config --global user.name "you" //사용자 이름 확인하기 위해 처음만 하면됨

이름으로 형상관리 시작하겠다

git config --global user.email "youjiyou82@gmail.com"

git add README.md // 한번만 하면 된다

git status // 매번 해줘야 함 상태 알기 위해

git add . // 추가하는 것

git commit -m "first commit" //10-18 (코드가 되면 -01 적어주기) 커밋 message :

왜 이거 작업하는지 이유가 적혀야 함

"10-18-01 중복코드제거"

git branch -M main

git remote add origin <https://github.com/youjiyou82/TDN.git>

// git init 과 같다 깃허브의 주소 최초 연결

(위에는 youjiyou82의 TDN에 저장하겠다는 것)

git push -u origin main // 위에가 main이니까 아래도 main인 것, Push하겠다

해당하는 데이터 올릴 것이다

\* 깃에 대해서 올리는 것이 끝이다

\* 어떠한 형태든 상관 없다 100mb 넘지 않으면 된다

* Git config –-global user.name과 email설정은 사용자설정으로 처음에 필수로 해야하는 것

텍스트, 스크린샷, 컴퓨터, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* Git status를 해주는 것은 상태변화를 확인하는 것인데 이 때 변경된 것이 있으면 변경된 파일이 빨간색으로 표시된다.
* Git add . // .은 모든 것을 추가해주겠다는 것

Git add . 를 해주면 변경된 사항을 추가해주겠다는 것이고, 다시 git status를 해주면 반영이 완료되었다는 것을 초록색으로 표시해준다

텍스트, 스크린샷, 소프트웨어, 컴퓨터이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

자동 생성된 설명

* Git commit -m “10-18” //message 적는 곳 지금은 “10-18”로 적었다

Commit 해주는 순간 git local에 저장됨

텍스트, 전자제품, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* 파일만들고 commit까지 한번에 했다

텍스트, 전자제품, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

텍스트, 디스플레이, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* rd 폴더명 /s //폴더명입력한 폴더를 비어있지않아도 삭제한다는 명령어

텍스트, 디스플레이, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

* 지금까지 작업한 훈련일지를 직접 git-hub TDN 레파지토리에 올리기

1. Git-work안에 폴더 만들고 그곳에 업로드할 파일들을 넣어놓는다.
2. Git 생성(사용자이름,init,status,add . 등)
3. Commit 메시지 지정
4. Branch & push 하면 올라가진다

아래가 에러난 것은 TDN폴더가 아니라 exm의 폴더로 지정해줬기 때문 !

텍스트, 전자제품, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 소프트웨어, 컴퓨터이(가) 표시된 사진

자동 생성된 설명

<git-hub>

Git-hub를 구글아이디로 로그인 하고 처음 레파지토리를 만든다

위에 cmd로 파일들 올리는 방법은 git-hub에 올리는 방법이다

텍스트, 소프트웨어, 웹 페이지, 웹사이트이(가) 표시된 사진

자동 생성된 설명

처음 git-hub의 레파지토리(저장소)를 만들면 아래와 같이 코드들을 참고할 수 있다

아래 드래그 한 부분은 commit에서 해주면 된다

드래그 안된 부분이 github로 올리는 코드

텍스트, 소프트웨어, 웹 페이지, 웹사이트이(가) 표시된 사진

자동 생성된 설명

드래그 안된 부분들 입력을 하면 아래와 같이 로그인 하는 창이 뜨고, 연결을 하면 commit한 파일들이 git-hub에 올라가져 있는 것을 확인 할 수 있다

텍스트, 스크린샷, 소프트웨어, 컴퓨터 아이콘이(가) 표시된 사진

자동 생성된 설명

텍스트, 소프트웨어, 웹 페이지, 컴퓨터 아이콘이(가) 표시된 사진

자동 생성된 설명  
github에 올리는 과정 (사용자지정하는 부분 제외한 전체부분)

텍스트, 전자제품, 스크린샷, 컴퓨터이(가) 표시된 사진

자동 생성된 설명