**Linux & git 보고서**

20기 인턴 강은구

목차

1. 실습 환경
   1. 사용한 pc(레노버 모델)
   2. Os 버전
   3. 네트워크 환경
2. 우분투 기본 명령어 학습
   1. Pwd, ls, cd, mkdir, rm, cp, mv, cat, nano/vim
   2. tree출력
3. 네트워크 및 패키지 관리
   1. Net0tools, ifconfig/ip a, ping
   2. Apt update/upgrade/install
4. 추천 프로그램 설치
   1. Terminator, gree
   2. VSCode, Chrome, GitHub Desktop, KakaoTalk
5. Git 설치 및 초기 설정
   1. Git 설치 확인
   2. User.name, user.email 설정
   3. Personal Access Token생성
6. Git 기본 사용법
   1. Git init, add, commit, push, pull
   2. 원격 저장소 연결
7. 폴더 및 파일 구조 정리
   1. Submodule 개념
   2. Submodule 추가/업데이트

**-실습 환경**

레노버 노트북에 Ubuntu 22.04 LTS르 설치하여 진행하였습니다. 설치 과정으 ㄴ듀얼 부팅 환경으로 구성되었으며, 파티션을 직접 분할하여 Linux 전역 공간을 확보하였습니다. 설치 후 기본 언어는 영어로 하였고 시간대는 서울로 설정하였습니다.

하드웨어 사양:

-노트북 모델: Lenovo

-CPU: Intel Core i7

-RAM: 24GB

-저장장치: 파티션 분할 300GB

-네트워크: 무선 LAN 사용

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

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

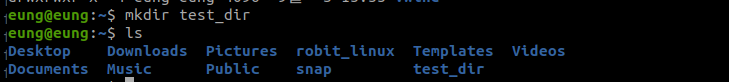
**-우분투 기본 명령어 학습**

Ubuntu 환경에서 가장 기본적인 작업은 터미널을 통해 명령어를 입력하여 파일과 디렉토리를 관리하는 것입니다. 보고서는 주로 디렉토리 이동, 파일 생성 및 삭제, 내용 확인 등의 기초 명령어를 중심으로 구성하였습니다.

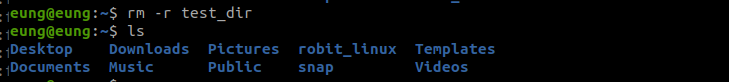
1. 디렉토리 관련 명령어



* 1. pwd: 현재 작업 중인 디렉토리 경로를 출력
  2. ls –al: 현재 디렉토리에 있는 파일과 폴더를 상세히 보여줌. 숨김 파일도 함께 표시
  3. cd[경로]: 지정한 경로로 디렉토리를 이동
  4. mkdir [폴더명]: 새로운 디렉토리를 생성



* 1. rm –r [폴더명]: 지정한 디렉토리와 그 안의 모든 파일을 삭제



1. 파일 관련 명령어
   1. touch [파일명]: 새로운 빈 파일을 생성
   2. cp [원본] [복사본]: 파일을 복사
   3. mv [원본] [새 이름]: 파일의이름을 변경하거나 다른 위치로 이동
   4. cat [파일명]: 파일의 내용을 터미널에 출력
   5. nano [파일명]/ vim [파일명]: 텍스트 파일을 편집
2. 디렉토리 구조 확인
   1. tree: 전체 디렉토리 구조를 트리 형태로 보여줌. (설치 필요: sudo apt instsall tree)

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

**-네트워크 및 패키지 관리**

Ubuntu 환경에서 네트워크 연결 상태를 확인하고 필요한 패키지를 설치/업데이트 하는 과정은 필수적입니다.

1. 네트워크 확인
   1. ifconfig: net\_tools 설치 후 사용 가능하며, 네트워크 인터페이스 정보(IP주소, MAC주소 등) 을 보여줌.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

* 1. ip a: 현재 시스템의 모든 네트워크 인터페이스 상태를 출렭

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

* 1. ping [주소]: 외부 서버와의 연결 상태를 테스트함. -> 예를 틀어 ping 8.8.8.8을 실행하면 구글 DNS 서버와 연결 여부를 확인 가능

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

1. 패키지 관리: Ubuntu에서 패키지는 apt 명령어로 설치 및 관리합니다.
   1. sudo apt update: 패키지 목록을 최신 상태로 갱신
   2. sudo apt upgrade: 설치된 패키지를 최신 버전으로 업데이트
   3. sudo apt install [패키지명]: 새로운 패키지를 설치
2. 패키지 설치 후 확인
   1. net\_tools: ifconfig 사용 가능
   2. tree: 디렉토리 구조 확인 가능
   3. terminator: 다중 창 터미널 실행 가능

**-추천 프로그램 설치**

Ubuntu 설치 직후에는 개발과 실습에 필요한 기본 프로그램들이 부족하기 때문에, 학습과 과제 진행에 도움되는 필수 프로그램들을 설치했습니다.

1. 필수 유틸리티(apt로 설치)
   1. Terminator: 다중 창 터미널 프로그램으로, 여러 개의 터미널을 한 화면에서 분할하여 사용할 수 있음. Git 명령어를 여러 창에서 동시에 실행하거나, 시스템 모니터링과 코드실행을 병행할 때 편리함.
   2. Tree: 디렉토리 구조를 트리 형태로 출력해주는 도구로, 과제 제출 시 폴더 구조 확인에 유용하게 활용
   3. Net\_tools: ifconfig, netstat 등의 네트워크 도구를 제공하며, 네트워크 상태 점검 시 사용
2. 개발 환경
   1. Visual studio Code(VSCode)
      1. 공식 사이트에서 ldeb파일을 다운받차 설치
      2. 명령어: sudo dpkg –i code\_\*.deb
      3. 다양한 확장 기능을 지원하여 개발에 필수적인 도구로 활용
   2. Google Chrome
      1. 공식 사이트에서 .deb 파일을 다운로드 받아 설치
      2. 웹 브라우징 및 GitHub 접근, 강의 자료 확인을 위해 사용
3. 협업/버전 관리 도구
   1. GitHub Desktop
      1. GitHub에서 제공하는 GUI기반 버전 관리 도구
      2. 로컬 Git 저장소와 GitHub 원격 저장소를 연동할 때, 명령어 기반보다 직관적으로 확인 가능
      3. 단, 최신 버전에서는 한글 지원 문제가 있어 이전 버전을 사용하는 것이 안정적임.
4. 기타 프로그램
   1. KakaoTalk(Wine 환경 활용)
      1. 공식 윈도우 설치 파일을 다운로드 하여 Wine을 통해 실행
      2. Ubuntu 환경에서 카카오톡을 실행할 수 있게 되어, 협업 시 메신저로 활용 가능

**-Git 설치 및 초기 설정**

1. Git 설치: Git은 리눅스에서 가장 널리 사용되는 버전 관리 도구로, 소스코드 이력을 관리하고 협업을 지원합니다. Ubuntu 22.04에서는 기본 패키지 관리잘를 이용해 간단하게 설치할 수 있습니다

>>sudo apt update

>>sudo apt install git –y

//설치 후 버전 확인

>>git –version

Git viersion 2.34.1(예시)

1. Git은 커시을 기록할 때 작성자의 이름과 이메일 주소를 함께 거장합니다. 따라서 최초 1회 사용자 정보를 설정해야합니다.

//설정된 정보 확인

>>git config –list

user.name=”eungoo”

[user.email=dmsrn@gmail.com](mailto:user.email=dmsrn@gmail.com)

-> 해당 이메일을 GitHub 계정에 등록된 이메일 주소와 동일해야 GitHub에서 커밋이 본인 계정과 연결됩니다. 원치 않을 경우 GitHub에서 제공하는 no-reply 이메일을 사용할 수도 있습니다.

1. Personal Access Token 생성: GitHub는 2021년 이후 보안 정책 강화로 인해 HTTPS인증 시 비밀번호 대신 Personal Access Token을 사용합니다. 토큰은 GitHub 웹사이트에서 다음과 같이 생성합니다.
   1. GitHub 로그인 -> Settings -> Developer Settings \_> Personal access tokens -> Tokens(classic)
   2. “Generate new token” 클릭 후 repo 권한 체크
   3. 발급된 토큰을 복사하여 안전한 곳에 저장

-> 해당 토큰은 GitHub에 push/pull을 할 때 비밀번호 대신 사용됩니다.

**-Git 기본 사용법**

Git은 코드 및 문서의 버전을 관리하기 위한 도구로, 로컬 저장소와 원격 저장소 간 동기화를 통해 협업과 이력 관리를 지원합니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

1. 로컬 저장소 생성

>>mkdir ~/git\_demo

>>cd ~/git\_demo

>>git init

Initialized empty Git repository in /home/eungoo/git\_demo/ .git/

1. 파일 추가 및 커밋: README 파일을 생성하고 변경 사항을 스테이징 후 커밋합니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.



1. 원격 저장소 연결: GitHub에 미리 생성한 저장소를 원격 저장소(origin)으로 등록합니다

>>git remote add origin [https://github.com/<username>/<repo>.git](https://github.com/%3cusername%3e/%3crepo%3e.git)

>>git branch –M main

1. 원격 저장소 업로드(push): 로컬 커밋을 GitHub 원격 저장소에 업로드합니다.

>>git push origin main

1. 원격 저장소에서 내려받기(pull): GitHub저장소에서 최신 변경사항을 로컬 저장소로 가져옵니다.

>>git pull origin main

1. 명령어 요약
   1. git init: 새로운 Git 저장소 생성
   2. git add<파일>: 변경된 파일을 스테이징
   3. git commit –m"메시지”: 스테이징된 파일을 커밋
   4. git remote add origin <URL>: 원격 젖아소 연결
   5. git push origin main: 변경사항 업로드
   6. git pull origin main: 원격 변경사항 가져오기

**-폴더 및 파일 구조 정리**

1. 최종 폴더 구조

>>tree ~/robit\_linux

1. 구성 설명
   1. Hw1/: 보고서 제출용 폴더
      1. Hw1\_report.docx: Linux 및 Git 학습 보고서(과제 1 제출용 문서\_
   2. Hw2/: 터미널 실습 및 영상 제출용 폴더
      1. README.md: 실습 기록 및 설명 파일
      2. docs/memo.txt: 문서 폴더에 포함된 텍스트 파일
      3. src/example.sh: 실행 가능한 예제 스트립트(출력: Hello HW2)
2. Submodule: 하나의 Git 저장소 안에서 다른 Git 저장소를 하위 폴더로 포함할 수 있는 기능-> 단순 파일 복사/ 붙여넣기가 아니라, 해당 라이브러리의 독립적인 Git 이력을 유지하면서 내 프로젝트에서 함께 관리할 수 있음. 텍스트, 스크린샷, 폰트이(가) 표시된 사진

   AI 생성 콘텐츠는 정확하지 않을 수 있습니다.텍스트, 스크린샷, 폰트이(가) 표시된 사진

   AI 생성 콘텐츠는 정확하지 않을 수 있습니다.
   1. Submodule과 일반 폴더 차이
      1. Submodule 폴더는 단순한 파일 집합이 아니라 특정 커밋을 가리키는 포인터로 저장됨
      2. 따라서 메인 프로젝트에서 submodule은 어느 시점의 버전으르 가리키고 있는 상태