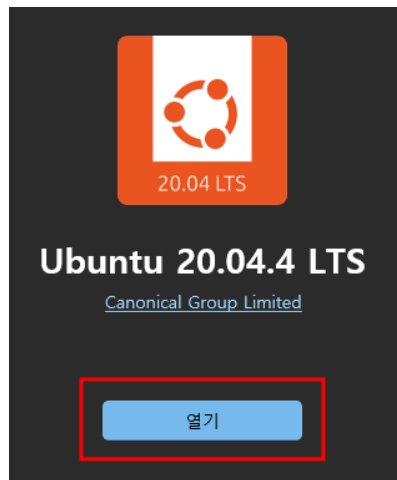
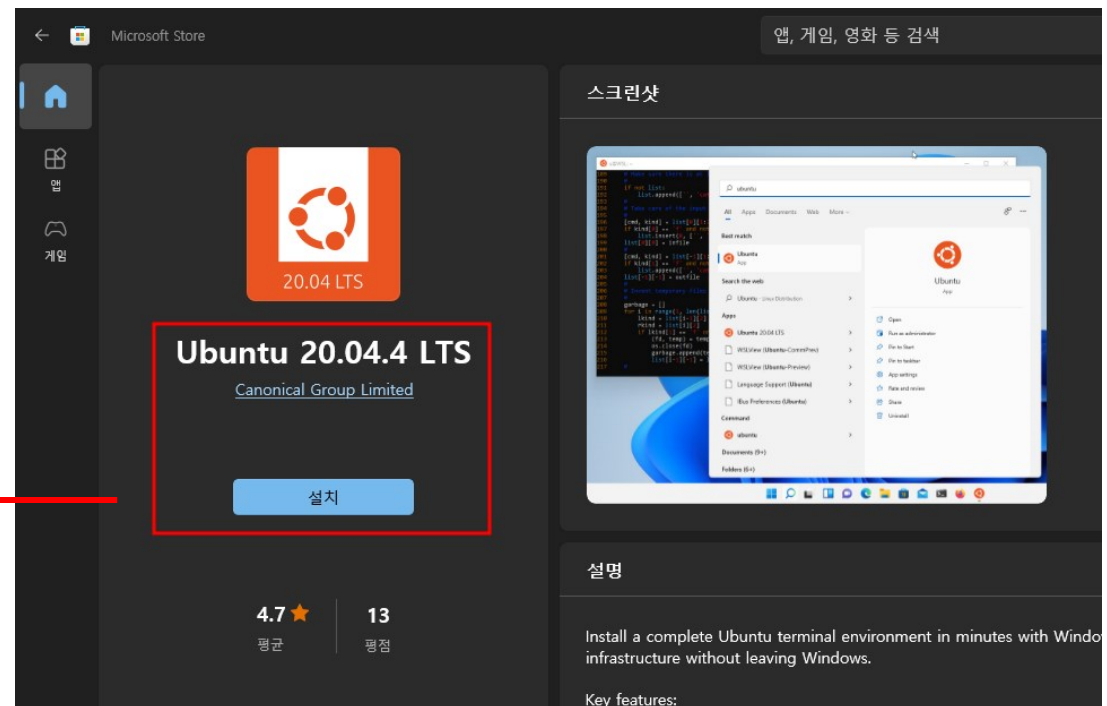


# WSL Ubuntu 20.04 설치 (1)

- **WSL 이란? (Windows Subsystem for Linux)**
  - Windows 환경에서 가상화를 통해 Linux 명령어, 프로그램을 실행할 수 있게 해주는 layer
- 설치 방법
  - Windows 검색 – Microsoft Store 실행
  - **Ubuntu 20.04 LTS** 설치



'열기' 클릭

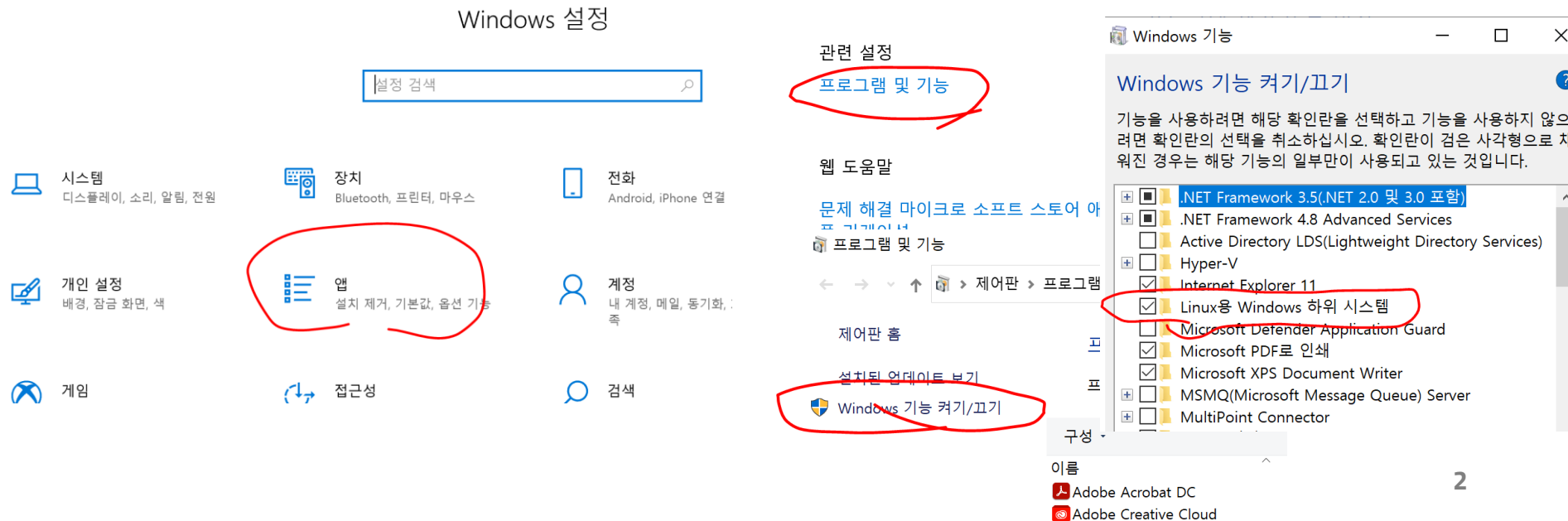


# 윈도우 하위 시스템 활성화

- 앞에서 우분투 실행시 아래와 같은 에러 발생시

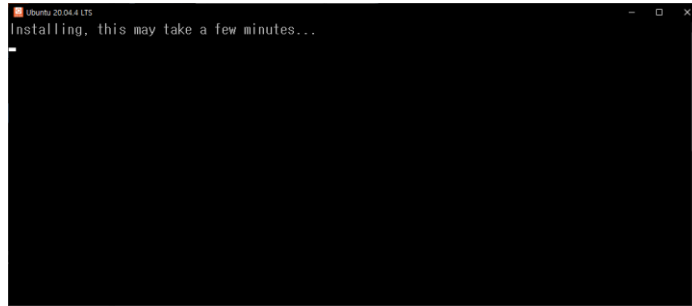
```
선택 Ubuntu 20.04.4 LTS
Installing, this may take a few minutes...
WslRegisterDistribution failed with error: 0x8007019e
The Windows Subsystem for Linux optional component is not enabled. Please enable it and try again.
See https://aka.ms/wslinstall for details.
Press any key to continue...
```

- 시작/제어판/앱/프로그램 및 기능/Windows 기능 켜기/끄기/Linux용 Windows 하위 시스템

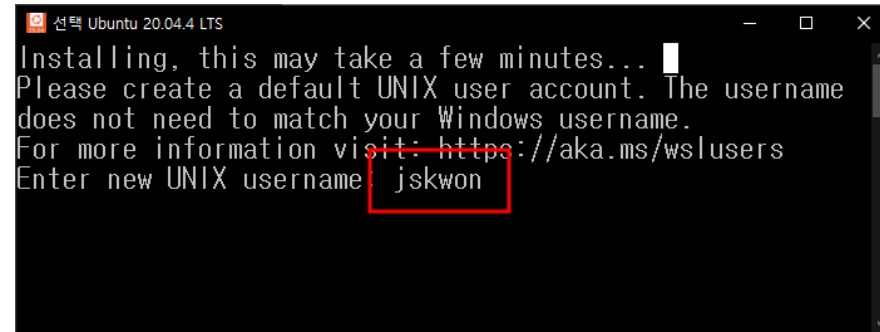


# WSL Ubuntu 20.04 설치 (2)

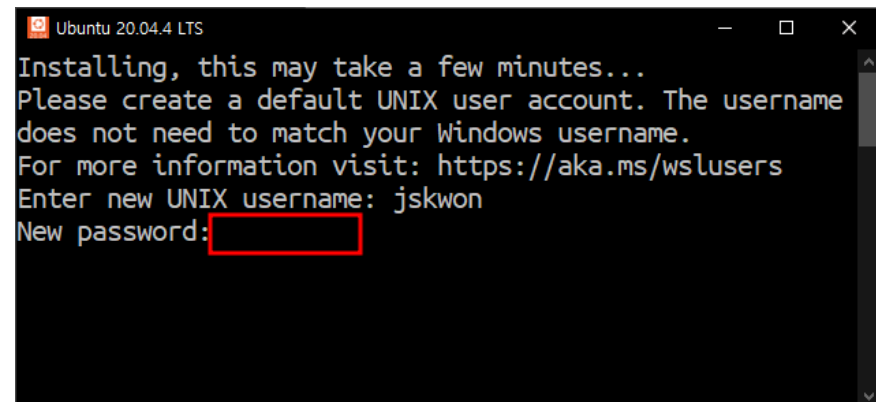
- 설치중...



- 계정 이름 입력 후 enter
  - 최초 WSL사용시 신규 ID
  - 기존에 이미 만들었으면 해당 아이디 입력



- 비밀번호 설정 후 enter
  - 비밀번호 2회 입력



# WSL Ubuntu 20.04 설치 (3)

- 로그인 완료 화면

```
선택 jskwon@jskwon-ThinkPadX1: ~
See "man sudo_root" for details.

Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 4.4.0-19041-Microsoft x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Mon Aug 29 11:53:03 KST 2022

System load:  0.52      Processes:            7
Usage of /home: unknown  Users logged in:      0
Memory usage: 46%      IPv4 address for eth1: 192.168.56.1
Swap usage:   0%       IPv4 address for wifi0: 192.168.1.182

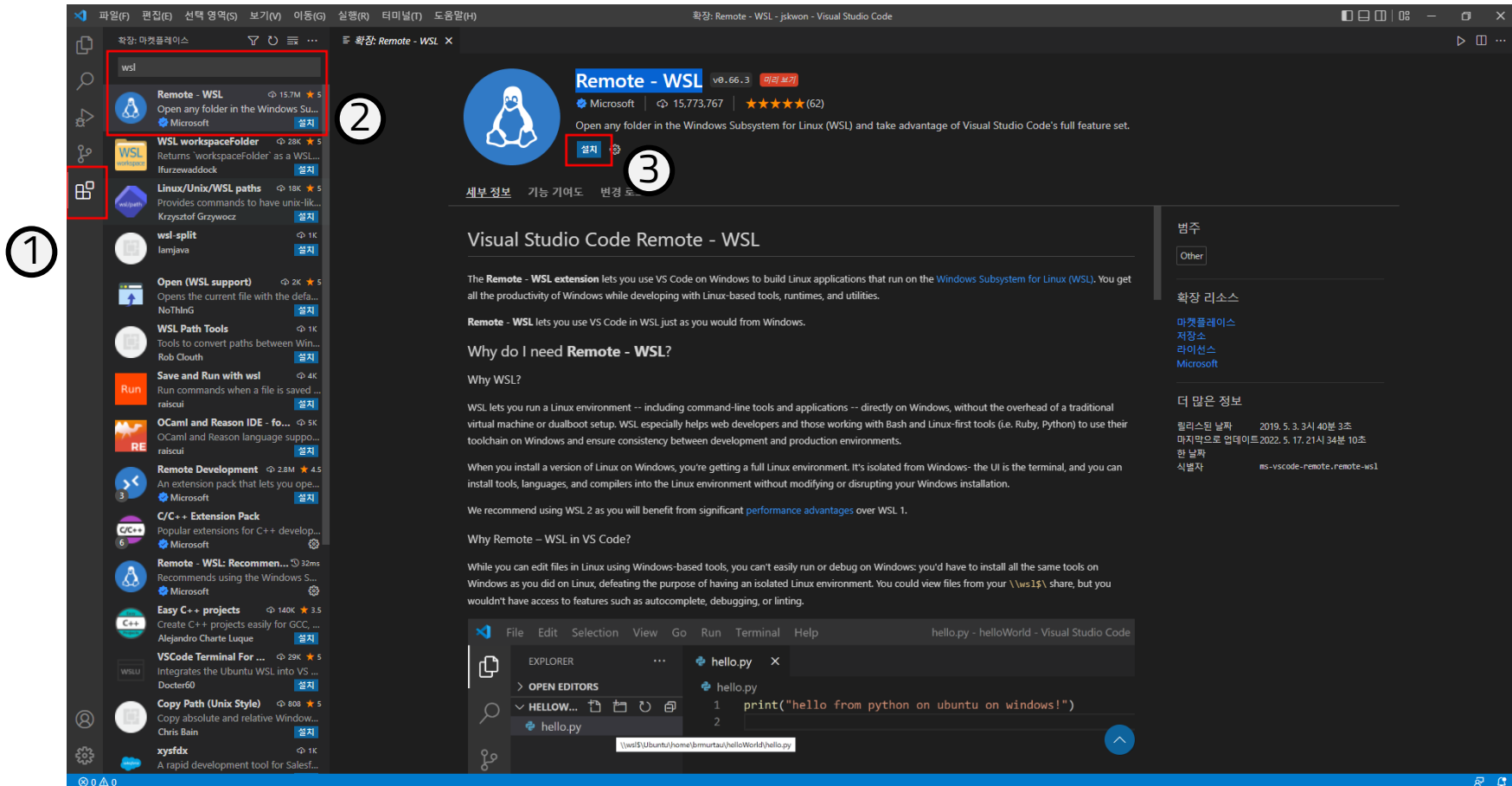
1 update can be applied immediately.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

This message is shown once a day. To disable it please create the
/home/jskwon/.hushlogin file.
jskwon@jskwon-ThinkPadX1:~$
```

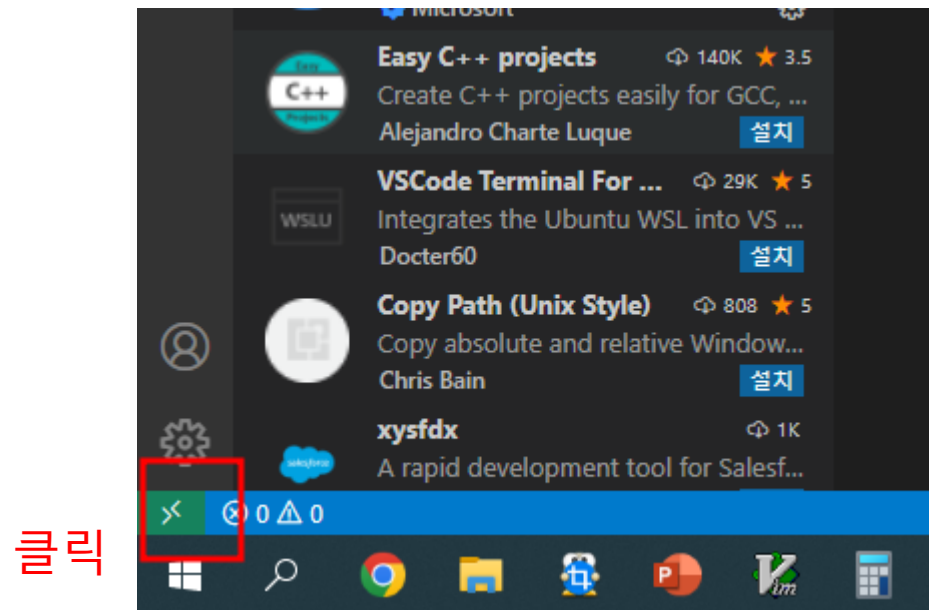
# WSL VSCode 연동 (1)

- Windows에 Visual Studio Code 설치
  - <https://code.visualstudio.com/download>
- Remote – WSL 확장 설치



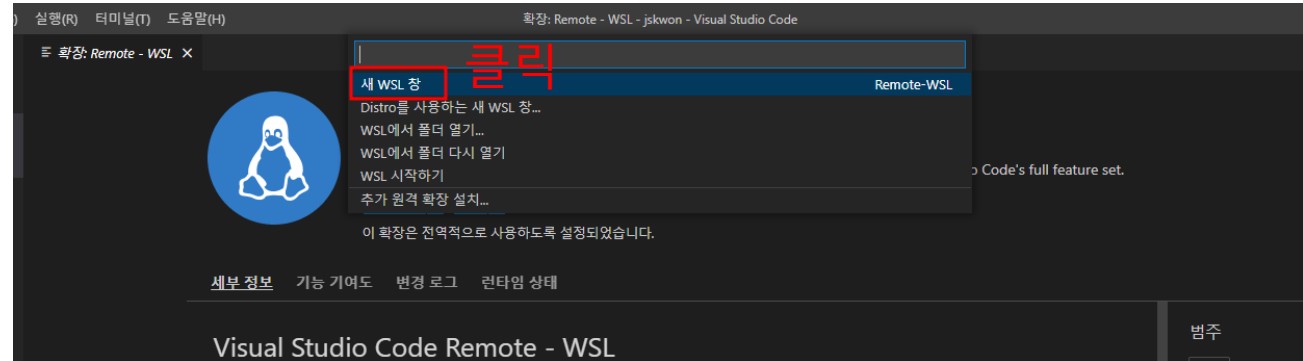
# WSL VSCode 연동 (2)

- **Remote – WSL 확장 설치 확인**
  - Visual Studio Code 좌측 하단에 초록색 WSL 버튼 생성



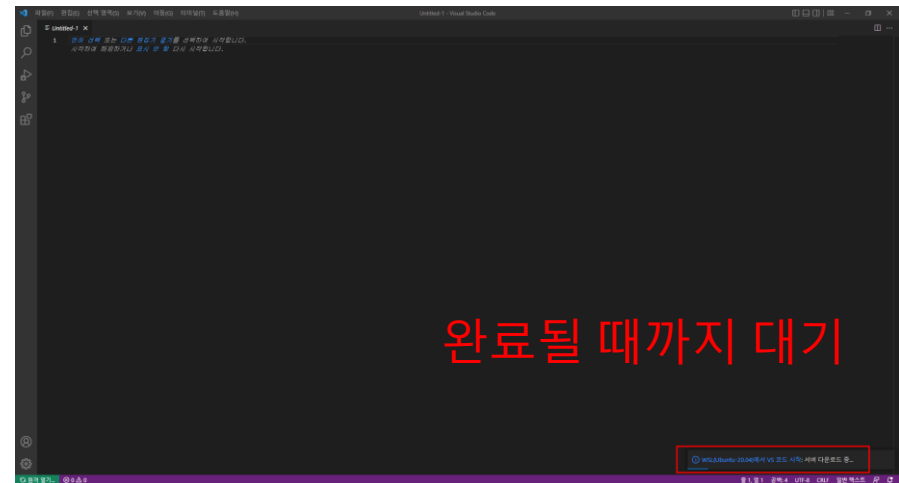
# WSL VSCode 연동 (3)

- 새 WSL 열기



- 새로 열린 WSL 창

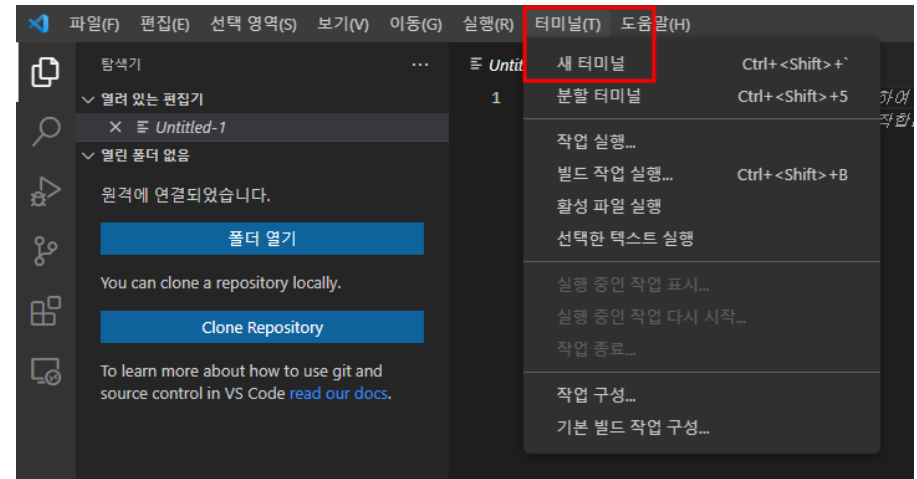
- WSL에 연결하여 VSCode를 새로 열기
- 이제 VSCode의 open/compile/save 대상은 피시가 아니라 WSL subsystem임



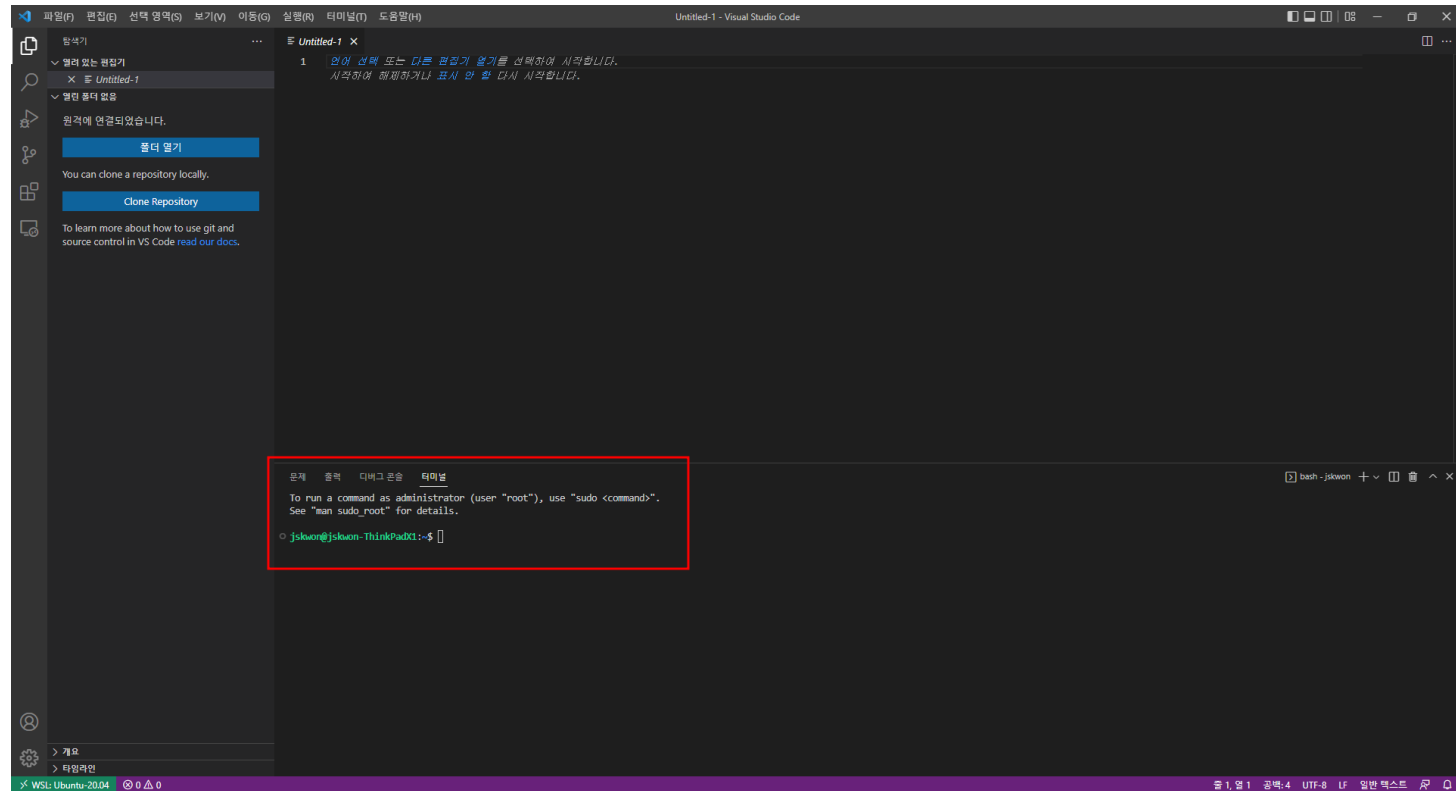
완료될 때까지 대기

# WSL VSCode 연동 (4)

- WSL 터미널 열기
  - 상단 메뉴 - 터미널 - 새 터미널



- 열린 터미널 창





# Embedded C 강의 코드 다운로드

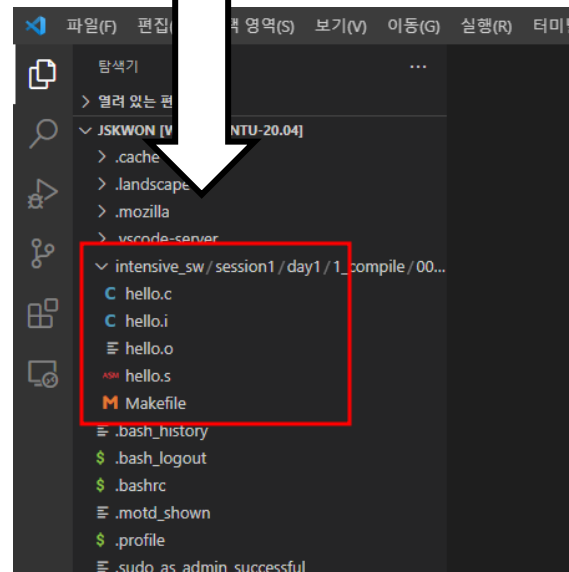
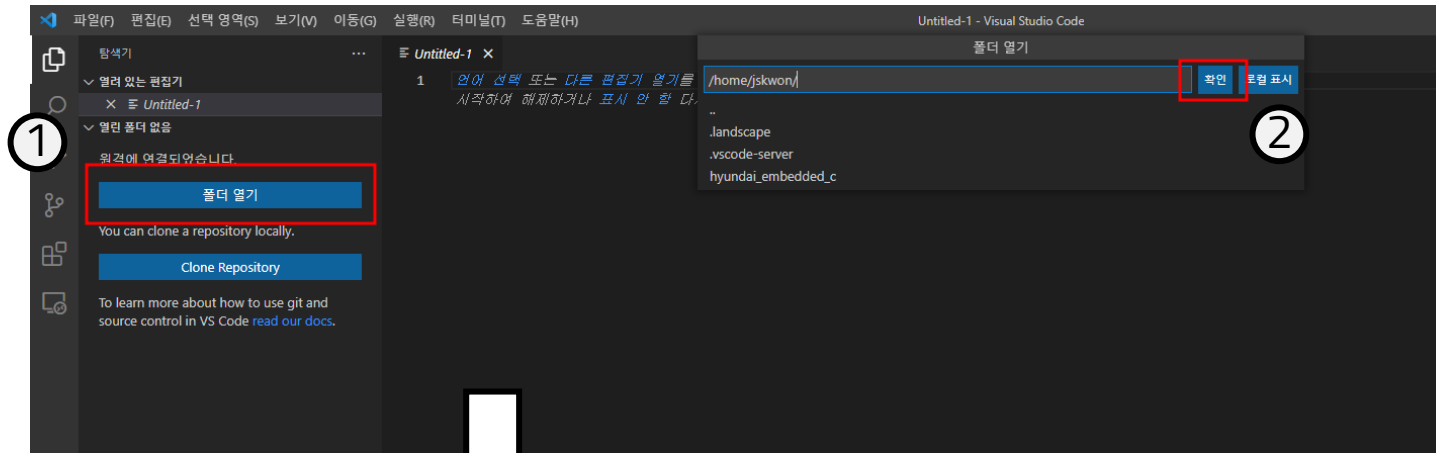
- Git 에 업로드 되어 있는 Embedded C 교육 코드 자료 clone
  - \$ sudo apt update
  - \$ sudo apt install build-essential -y
  - \$ git clone https://github.com/AI-SoC/intensive\_sw.git
  - \$ cd intensive\_sw
  - \$ ls

```
문제   출력   디버그 콘솔   터미널

● jskwon@jskwon-ThinkPadX1:~$ git clone https://github.com/AI-SoC/intensive_sw.git
Cloning into 'intensive_sw'...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 11 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (11/11), 3.88 KiB | 43.00 KiB/s, done.
● jskwon@jskwon-ThinkPadX1:~$ cd intensive_sw/
● jskwon@jskwon-ThinkPadX1:~/intensive_sw$ ls
session1
○ jskwon@jskwon-ThinkPadX1:~/intensive_sw$
```

# Embedded C 강의 코드 폴더 열기

- Visual Studio Code 내에서 코드가 저장된 폴더 열기



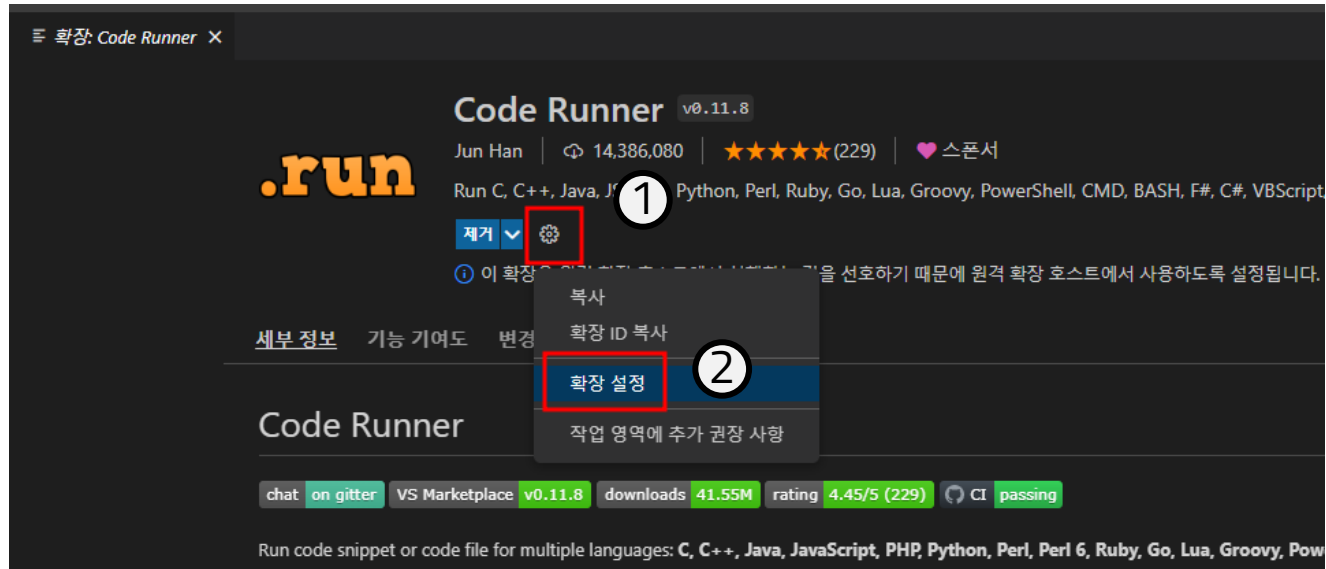
# Embedded C 강의 코드 컴파일 (1)

- VSCode 화면 폰트 크기 변경
  - 확대: ctrl + '+'
  - 축소: ctrl + '-'
- Make 툴을 사용한 컴파일 VSCode 연동
  - VSCode 확장에서 Code Runner 검색 및 설치 (WSL: Ubuntu-20.04에 설치 클릭)



# Embedded C 강의 코드 컴파일 (2)

- Code Runner 확장 설정



# Embedded C 강의 코드 컴파일 (3)

- Makefile 연동 설정

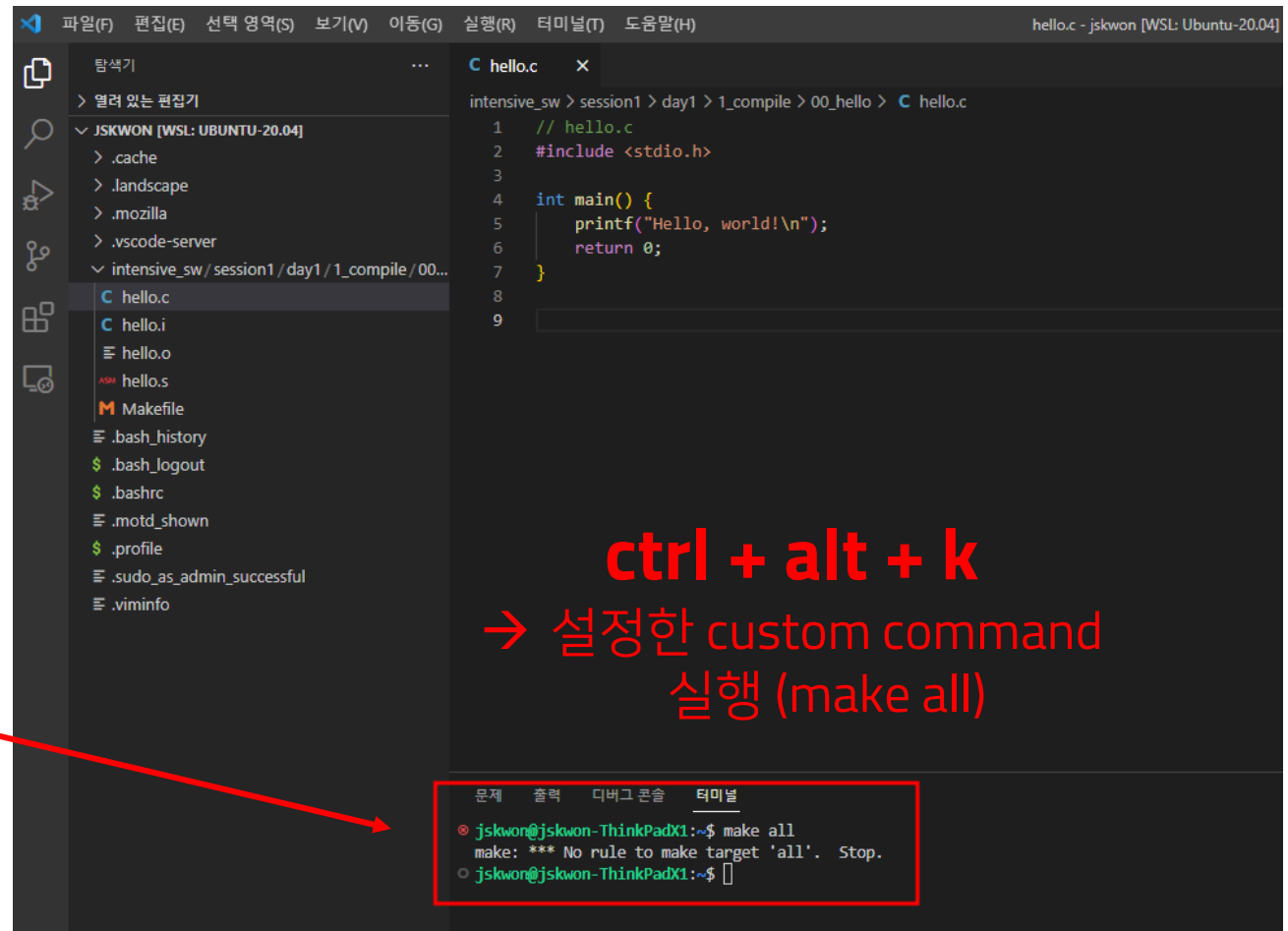
설정한 custom command를  
실행하는 명령어  
→ **ctrl + alt + k**



# Embedded C 강의 코드 컴파일 (4)

- "make all" command 실행

최초 command 실행하면  
기본 home (~)  
디렉토리에서 실행됨  
코드가 있는 경로로  
이동해야 함



The screenshot shows the Visual Studio Code interface. On the left, the Explorer panel shows the file structure of a project in WSL. The file `hello.c` is selected. The main editor shows the contents of `hello.c`, which includes a standard C program with `printf`. The bottom panel shows the Terminal window. A red box highlights the terminal output, which shows an error when running `make all`: `make: *** No rule to make target 'all'. Stop.`. A red arrow points from the text '이동해야 함' to the terminal window.

```
intensive_sw > session1 > day1 > 1_compile > 00_hello > C hello.c
1 // hello.c
2 #include <stdio.h>
3
4 int main() {
5     printf("Hello, world!\n");
6     return 0;
7 }
8
9
```

**ctrl + alt + k**  
→ 설정한 custom command  
실행 (make all)

```
문제 출력 디버그 콘솔 터미널
* jskwon@jskwon-ThinkPadX1:~$ make all
make: *** No rule to make target 'all'. Stop.
o jskwon@jskwon-ThinkPadX1:~$
```

# Embedded C 강의 코드 컴파일 (5)

- 컴파일 대상 코드가 존재하는 폴더 위치로 이동

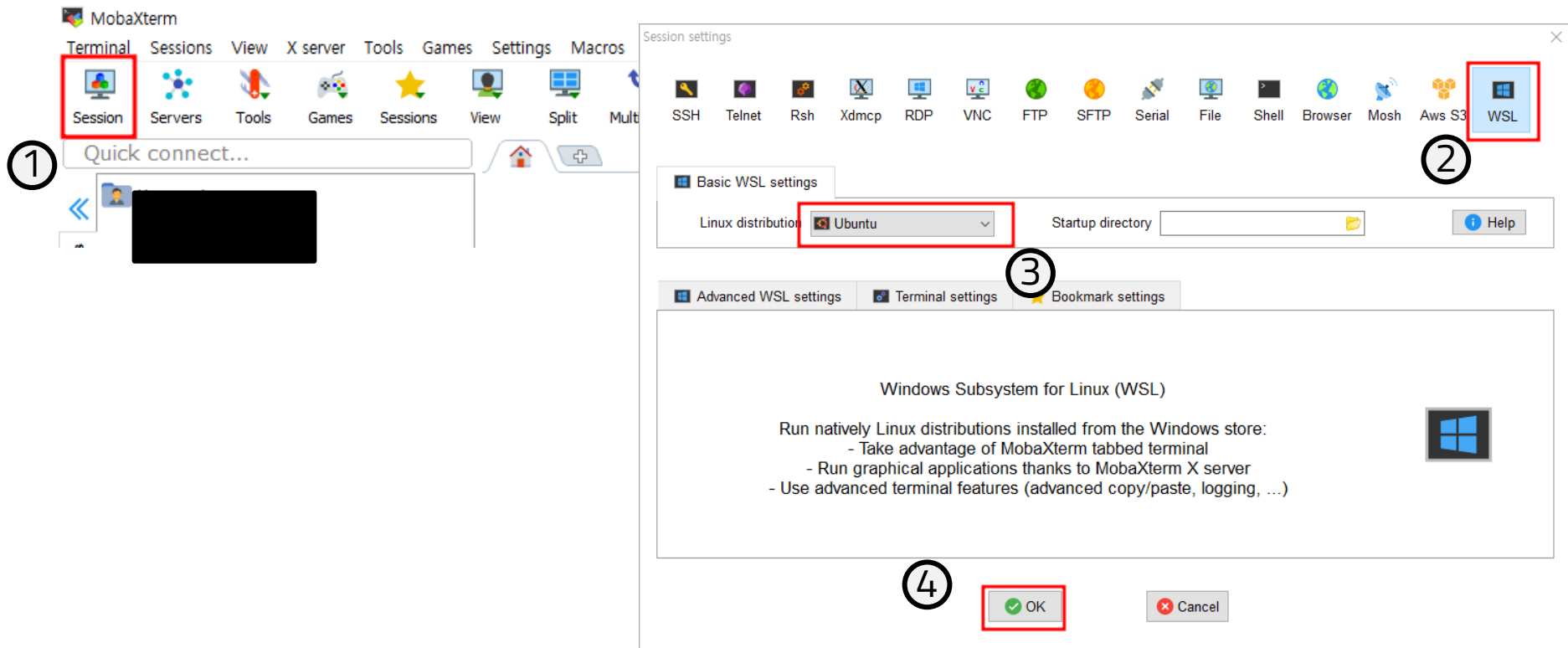
```
intensive_sw > session1 > day1 > 1_compile > 00_hello > C hello.c
1 // hello.c
2 #include <stdio.h>
3
4 int main() {
5     printf("Hello, world!\n");
6     return 0;
7 }
8
9
```

```
문제 출력 디버그 콘솔 터미널
1
jshwon@jshwon-ThinkPadX1:~$ make all
make: *** No rule to make target 'all'. Stop.
jshwon@jshwon-ThinkPadX1:~$ cd intensive_sw/session1/day1/1_compile/00_hello/
jshwon@jshwon-ThinkPadX1:~/intensive_sw/session1/day1/1_compile/00_hello$ make all
gcc -o hello hello.o
/usr/bin/ld: hello.o:hello.c:(.text+0x9): undefined reference to `__main'
collect2: error: ld returned 1 exit status
make: *** [Makefile:21: all] Error 1
jshwon@jshwon-ThinkPadX1:~/intensive_sw/session1/day1/1_compile/00_hello$
```

이후에 다시 명령어 입력  
ctrl + alt + k

# WSL GUI 사용하기 (1)

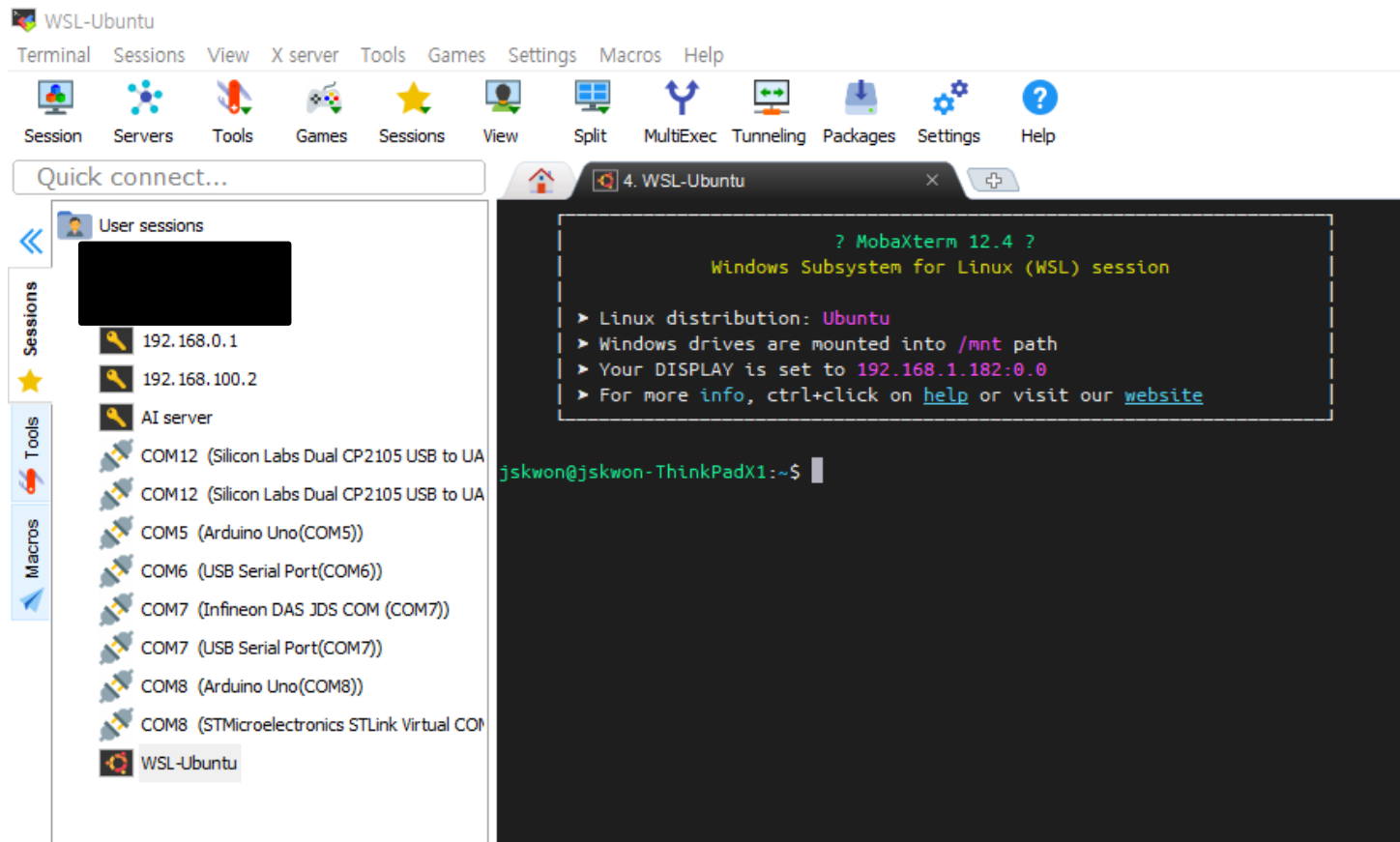
- MobaXterm 다운로드 및 설치
  - <https://mobaxterm.mobatek.net/download.html>
- MobaXterm 실행하여 WSL session 열기





# WSL GUI 사용하기 (2)

- 최초 1번 연결만 되면 OK
- **GUI 프로그램 사용하려면 창 닫지 말고 background에 띄워 놓기**



# WSL GUI 사용하기 (3)

- GUI 테스트 – 인터넷 브라우저 (Firefox)
- 다음 명령어로 firefox 브라우저 설치
  - `$ sudo apt update`
  - `$ sudo apt install firefox fonts-nanum* -y`
- 브라우저 실행
  - `$ firefox`
- 만약 실행이 안된다면...
  - MobaXterm 켜져 있는지 확인

