

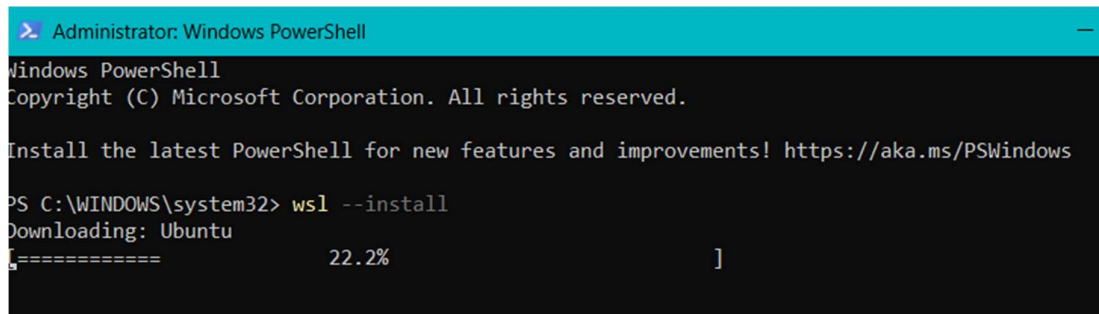
# Practical: Installation of Java on Unix/Linux Machine (Without VirtualBox)

■ This guide uses WSL (Windows Subsystem for Linux) to simulate a Linux environment directly on Windows.

Step 1: Enable WSL

1. Open PowerShell as Administrator
2. Run the following command:

`wsl --install`

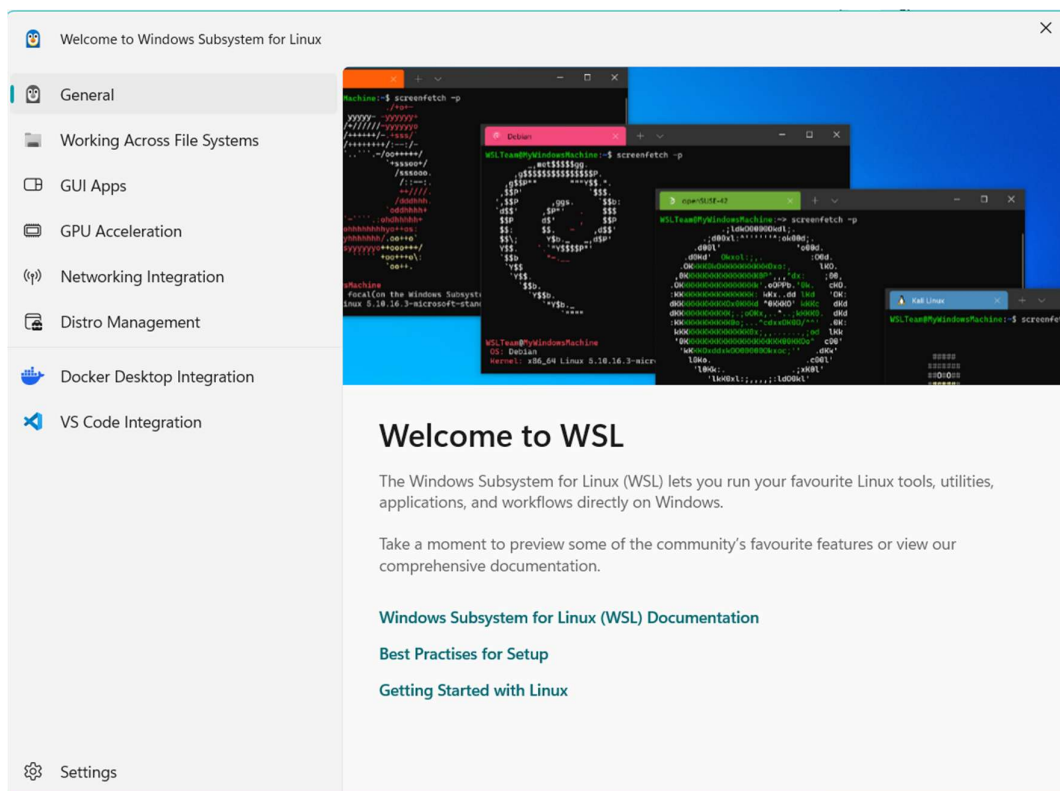


```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

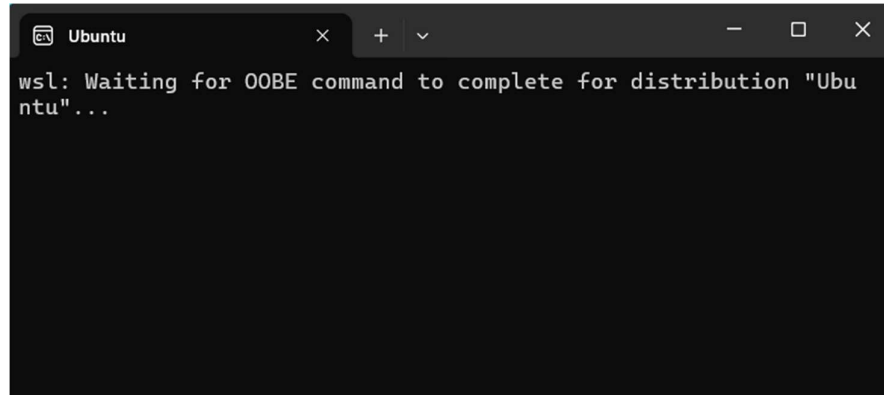
PS C:\WINDOWS\system32> wsl --install
Downloading: Ubuntu
===== 22.2% ]
```

3. Restart your computer when prompted.
4. After restart, choose Ubuntu or install it from the Microsoft Store.



## Step 2: Open Ubuntu (WSL)

- Search for 'Ubuntu' in Start Menu and open it.



- It will initialize and ask for a username and password.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\WINDOWS\system32> wsl --install
Downloading: Ubuntu
Installing: Ubuntu
Distribution successfully installed. It can be launched via 'wsl.exe -d Ubuntu'
Launching Ubuntu...
Provisioning the new WSL instance Ubuntu
This might take a while...
Create a default Unix user account: Shivangi
```

## Step 3: Update the Package List

Run the following command:

`sudo apt update`

```
shivangi@Shivangi:~$ sudo apt update
[sudo] password for shivangi:
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1065 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [185 kB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21.6 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [879 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [194 kB]
Get:9 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.3 kB]
Get:11 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [17.0 kB]
Get:12 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [1587 kB]
Get:13 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [350 kB]
Get:14 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:15 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [18.5 kB]
Get:16 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [4288 B]
Get:17 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:18 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [380 B]
```

## Step 4: Install Java (OpenJDK 8 or 11)

- For Java 8:

`sudo apt install openjdk-8-jdk -y`

```
shivangi@Shivangi:~$ sudo apt install openjdk-8-jdk -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  alsa-topology-conf alsa-ucm-conf ca-certificates-java fonts-dejavu-extra java-common libasound2-data libasound2t64 libasyns0 libatk-wrapper-java
  libatk-wrapper-java-jni libflac12t64 libgail-common libgail18t64 libgif7 libgtk2.0-0t64 libgtk2.0-bin libgtk2.0-common libice-dev libice6 libmp3lame0
  libmpeg2-3t64 libnss3 libogg0 libopus0 libpcsc-lite1 libpthread-stubs0-dev libpulse0 libsm-dev libsm6 libsndfile1 libvorbis0a libvorbisenc2
  libx11-dev libxau-dev libxaw7 libxcb-shape0 libxcb1-dev libxdmcp-dev libxft2 libxkbfile1 libxmu6 libxpm4 libxt-dev libxt6t64 libxv1 libxxf86dgal
  openjdk-8-jdk-headless openjdk-8-jre openjdk-8-jre-headless x11-utils x11proto-dev xorg-sgml-doctools xtrans-dev
Suggested packages:
  default-jre alsa-utils libasound2-plugins gvfs libice-doc opus-tools pcsd pulseaudio libsm-doc libx11-doc libxcb-doc libxt-doc openjdk-8-demo
  openjdk-8-source visualvm libnss-mdns fonts-nanum fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei fonts-wqy-zenhei fonts-indic mesa-utils
Recommended packages:
  luit
The following NEW packages will be installed:
  alsa-topology-conf alsa-ucm-conf ca-certificates-java fonts-dejavu-extra java-common libasound2-data libasound2t64 libasyns0 libatk-wrapper-java
  libatk-wrapper-java-jni libflac12t64 libgail-common libgail18t64 libgif7 libgtk2.0-0t64 libgtk2.0-bin libgtk2.0-common libice-dev libice6 libmp3lame0
  libmpeg2-3t64 libnss3 libogg0 libopus0 libpcsc-lite1 libpthread-stubs0-dev libpulse0 libsm-dev libsm6 libsndfile1 libvorbis0a libvorbisenc2
  libx11-dev libxau-dev libxaw7 libxcb-shape0 libxcb1-dev libxdmcp-dev libxft2 libxkbfile1 libxmu6 libxpm4 libxt-dev libxt6t64 libxv1 libxxf86dgal
  openjdk-8-jdk openjdk-8-jdk-headless openjdk-8-jre openjdk-8-jre-headless x11-utils x11proto-dev xorg-sgml-doctools xtrans-dev
0 upgraded, 55 newly installed, 0 to remove and 2 not upgraded.
Need to get 54.4 MB of archives.
After this operation, 183 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu noble/main amd64 alsa-topology-conf all 1.2.5.1-2 [15.5 kB]
```

- For Java 11:

sudo apt install openjdk-11-jdk -y

```
shivangi@Shivangi:~$ sudo apt install openjdk-11-jdk -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless
Suggested packages:
  openjdk-11-demo openjdk-11-source visualvm libnss-mdns fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei | fonts-wqy-zenhei fonts-indic
The following NEW packages will be installed:
  openjdk-11-jdk openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless
0 upgraded, 4 newly installed, 0 to remove and 2 not upgraded.
Need to get 118 MB of archives.
After this operation, 260 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 openjdk-11-jre-headless amd64 11.0.28+6-1ubuntu1~24.04.1 [42.3 MB]
7% [1 openjdk-11-jre-headless 9914 kB/42.3 MB 23%] 321 kB/s 5min 35s
```

Step 5: Verify Java Installation

Run:

java -version

```
shivangi@Shivangi:~$ java -version
openjdk version "11.0.28" 2025-07-15
OpenJDK Runtime Environment (build 11.0.28+6-post-Ubuntu-1ubuntu124.04.1)
OpenJDK 64-Bit Server VM (build 11.0.28+6-post-Ubuntu-1ubuntu124.04.1, mixed mode, sharing)
shivangi@Shivangi:~$ |
```

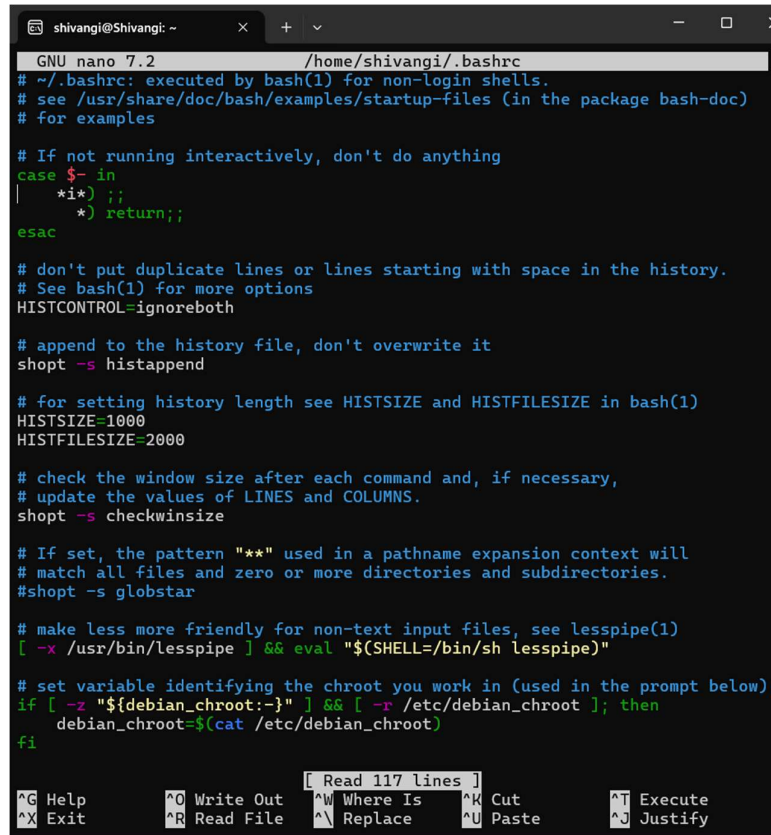
Expected output:

openjdk version "1.8.0\_xxx" ...

Step 6: Set JAVA\_HOME Environment Variable (Optional)

1. Open .bashrc file:

nano ~/.bashrc



```
GNU nano 7.2 /home/shivangi/.bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
| *i*) ;;
| *) return;;
esac

# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options
HISTCONTROL=ignoreboth

# append to the history file, don't overwrite it
shopt -s histappend

# for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
HISTSIZE=1000
HISTFILESIZE=2000

# check the window size after each command and, if necessary,
# update the values of LINES and COLUMNS.
shopt -s checkwinsize

# If set, the pattern "*" used in a pathname expansion context will
# match all files and zero or more directories and subdirectories.
#shopt -s globstar

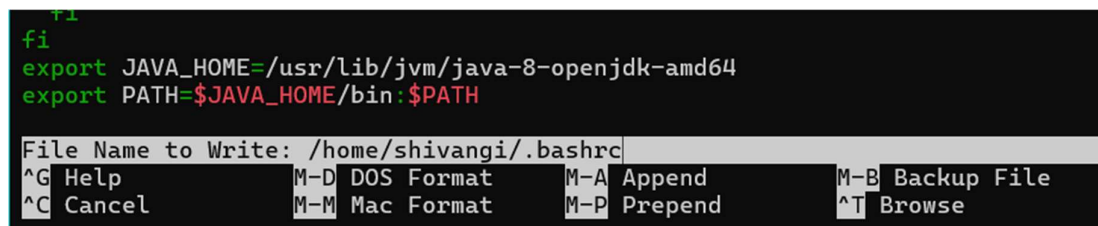
# make less more friendly for non-text input files, see lesspipe(1)
[ -x /usr/bin/lesspipe ] && eval "$(SHELL=/bin/sh lesspipe)"

# set variable identifying the chroot you work in (used in the prompt below)
if [ -z "${debian_chroot:-}" ] && [ -r /etc/debian_chroot ]; then
    debian_chroot=$(cat /etc/debian_chroot)
fi
```

2. Add the following lines at the end:

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
```

```
export PATH=$JAVA_HOME/bin:$PATH
```




```
fi
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export PATH=$JAVA_HOME/bin:$PATH
File Name to Write: /home/shivangi/.bashrc
^G Help      M-D DOS Format  M-A Append     M-B Backup File
^C Cancel    M-M Mac Format  M-P Prepend    ^T Browse
```

3. Save and exit (Ctrl+X, then Y, then Enter)

4. Reload bashrc: source ~/.bashrc

5. Verify JAVA\_HOME is set:

```
echo $JAVA_HOME
```



```
shivangi@Shivangi:~$ echo $JAVA_HOME
/usr/lib/jvm/java-8-openjdk-amd64
shivangi@Shivangi:~$
```

■ You now have Java installed on a Linux system simulated within Windows without using VirtualBox!