

DevOps Training-Day-1

Installing and Setting Up WSL with Ubuntu on Windows 10

Step 1: Enable WSL

Before installing Ubuntu, ensure that WSL is enabled on your Windows system.

Enable WSL Feature

1. Open **PowerShell** as Administrator and run:
2. `wsl --install`

This installs the default Linux distribution and enables necessary components.

3. If WSL is already installed but not enabled, use:
4. `dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart`
5. Enable the Virtual Machine Platform feature (required for WSL 2):
6. `dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart`
7. Restart your computer to apply changes.

Step 2: Install Ubuntu

1. Open **Command Prompt** or **PowerShell** and run:
2. `wsl --install -d Ubuntu`

If the installation fails due to timeout issues, retry the command after shutting down WSL:

```
wsl --shutdown
```

```
wsl --install -d Ubuntu
```

3. Once installed, start Ubuntu:
4. `wsl.exe -d Ubuntu`

Step 3: Set Up Ubuntu

When Ubuntu runs for the first time, it will ask you to create a new user account.

1. **Enter a username** (must start with a lowercase letter or underscore, and contain only lowercase letters, digits, underscores, and dashes).
2. **Set a password** (enter and confirm the password). If passwords do not match, you will need to retry.
3. Once successful, Ubuntu will be set up and ready to use.

Step 4: Verify Installation

To check the installed distributions and their versions:

```
wsl -l -v
```

To verify Ubuntu is running:

```
wsl -d Ubuntu
```

Step 5: Configure Ubuntu

Update System Packages

After logging in, update the package list and upgrade installed packages:

```
sudo apt update && sudo apt upgrade -y
```

Set Default WSL Version

To use WSL 2 as the default version for future installations:

```
wsl --set-default-version 2
```

To check the current WSL version:

```
wsl -l -v
```

To convert an existing installation to WSL 2:

```
wsl --set-version Ubuntu 2
```

Step 6: Enable .hushlogin to Suppress Login Message

To disable the daily login message, create a .hushlogin file in your home directory:

```
touch ~/.hushlogin
```

Additional Commands

Restart WSL:

```
wsl --shutdown
```

Uninstall a Distribution:

```
wsl --unregister Ubuntu
```

Access Windows Files in WSL:

```
cd /mnt/c
```

Conclusion

You have successfully installed and set up WSL with Ubuntu on Windows 10. You can now use the Ubuntu terminal to run Linux commands and manage your system efficiently.

```
sachin@LAPTOP-KCFSOJ17:~ x + v

Provisioning the new WSL instance Ubuntu
This might take a while...
Create a default Unix user account: sachin
New password:
Retype new password:
passwd: password updated successfully
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 5.15.167.4-microsoft-standard-WSL2 x86_64)

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

System information as of Tue Mar 18 16:20:32 UTC 2025

System load: 0.66      Processes:          32
Usage of /:   0.1% of 1006.85GB  Users logged in:     0
Memory usage: 6%           IPv4 address for eth0: 172.22.255.38
Swap usage:   0%

This message is shown once a day. To disable it please create the
/home/sachin/.hushlogin file.
sachin@LAPTOP-KCFSOJ17:~$ sudo gpt update
[sudo] password for sachin:
sudo: gpt: command not found
sachin@LAPTOP-KCFSOJ17:~$ sudo gpt update -y
sudo: gpt: command not found
sachin@LAPTOP-KCFSOJ17:~$ sudo apt update -y
Hit:1 http://archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [670 kB]
Get:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Ign:6 http://archive.ubuntu.com/ubuntu noble/universe amd64 Packages
Get:7 http://archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [130 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [8964 B]
Get:10 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [6912 B]
```

```

sachin@LAPTOP-KCFSOJ17:~$ sudo apt update -y
[sudo] password for sachin:
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 Components [8964 B]
Get:5 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.0 kB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:7 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:8 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:9 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [921 kB]
Get:10 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [151 kB]
Get:11 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1040 kB]
Get:12 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [363 kB]
Get:13 http://archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:14 http://archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]
Get:15 http://archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:16 http://archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [20.0 kB]
Get:17 http://archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:18 http://archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Fetched 2937 kB in 6s (531 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
56 packages can be upgraded. Run 'apt list --upgradable' to see them.
sachin@LAPTOP-KCFSOJ17:~$ sudo apt install -y openjdk-17-jdk
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
alsa-topology-conf alsamixer-conf ca-certificates-java fonts-dejavu-extra java-common libasound2-data libasound2t64
libatk-wrapper-java libatk-wrapper-java-jni libgif7 libice-dev libice6 libnspr4 libnss3 libpcsc-lite1
libpthread-stubs0-dev libsm-dev libsm6 libx11-dev libxau-dev libxaw7 libxcb-shape0 libxcb1-dev libxdmpc-dev libxft2
libxkbfile1 libxmu6 libxpm4 libxt-dev libxt6t64 libxv1 libxf86dg1 openjdk-17-jdk-headless openjdk-17-jre
openjdk-17-jre-headless x11-utils x11proto-dev xorg-sgml-doctools xtrans-dev
Suggested packages:
default-jre alsamixer libasound2-plugins libice-doc pcscd libsm-doc libx11-doc libxcb-doc libxt-doc openjdk-17-demo
openjdk-17-source visualvm libnss-mdns fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei
| fonts-wayzenhei fonts-indic mesa-utils
Recommended packages:
luit

```

Step-by-Step Guide to Creating a Freestyle Job in Jenkins to Install Nginx on a Local Ubuntu VM

Prerequisites for Setting Up a Freestyle Job to Install Nginx in Jenkins

Before creating the Freestyle Job, ensure that the following prerequisites are met:

1. Install Jenkins on Ubuntu (If Not Installed)

If Jenkins is not installed on your Ubuntu VM, follow these steps:

Step 1: Update Package Lists

`sudo apt update -y`

Step 2: Install Java (Required for Jenkins)

`sudo apt install -y openjdk-17-jdk`

Step 3: Verify Java Version

`java -version`

Step 4: Add Jenkins Repository Key

(Note: The `apt-key add` command is deprecated in newer Ubuntu versions. Use the correct method below.)

Correct Way to Add Jenkins Repository (Without `apt-key`)

Step 4.1: Add Jenkins GPG Key

```
wget -q -O- https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee  
/usr/share/keyrings/jenkinskeyring.asc > /dev/null
```

Step 4.2: Add Jenkins Repository

```
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-  
stable binary/" |
```

```
sudo tee /etc/apt/sources.list.d/jenkins.list > /dev/null
```

Step 5: Install Jenkins

```
sudo apt update -y
```

```
sudo apt install -y jenkins
```

Step 6: Start and Enable Jenkins Service

```
sudo systemctl start jenkins
```

```
sudo systemctl enable jenkins
```

Step 7: Check Jenkins Status

```
sudo systemctl status jenkins
```

2. Access Jenkins Web Interface

Jenkins will be available at `http://<VM_IP>:8080`

To Get the Jenkins Server URL, Follow These Steps:

Method 1: Check the Default URL

By default, Jenkins runs on port 8080. Open in a browser:

`http://<your-server-ip>:8080`

If you're on the same machine as Jenkins, use:

`http://localhost:8080`

Method 2: Get Server IP Address

```
hostname -I
```

or

```
ip a | grep inet
```

Method 3: Check Jenkins Logs (If Unable to Access)

```
sudo journalctl -u jenkins --no-pager --lines=50
```

Look for lines mentioning "*Jenkins is fully up and running*" and the URL.

3. Access Jenkins Web Interface and Log In

1. Open a browser and go to `http://<JENKINS_SERVER_IP>:8080`
2. Enter the username (admin) and the admin password retrieved from the following command:

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

3. Choose *Install Suggested Plugins* (recommended) or manually select plugins.

4. Ensure Sudo Access for Jenkins User

Jenkins runs as a system user (jenkins). If your script requires sudo, allow Jenkins to execute commands without a password:

```
sudo visudo
```

Add the following line at the end of the file:

```
jenkins ALL=(ALL) NOPASSWD: ALL
```

Save and exit.

Step-by-Step Guide to Creating a Freestyle Job in Jenkins to Install Nginx

Step 1: Create a New Freestyle Job

1. Click on **New Item** from the Jenkins Dashboard.
2. Enter a name for the job, e.g., *Install-Nginx*.
3. Select **Freestyle project**.
4. Click **OK**.

Step 2: Configure the Job Add Build Step

1. Scroll down to **Build** → Click *Add build step* → Select **Execute shell**.
2. Paste the following script in the command box:

```
#!/bin/bash  
  
echo "Updating package lists..."  
sudo apt update -y  
  
echo "Installing Nginx..."  
sudo apt install -y nginx
```

```
echo "Starting Nginx service..."  
sudo systemctl start nginx  
echo "Enabling Nginx to start on boot..."  
sudo systemctl enable nginx
```

```
echo "Nginx Installation Completed!"
```

Step 3: Save and Run the Job

1. Click **Save**.
2. Click **Build Now**.
3. Check the **Console Output** to verify the installation.

Step 4: Verify the Installation

1. Check Nginx Status

```
systemctl status nginx
```

If running, you should see output like "*active (running)*".

2. Open Nginx in Browser

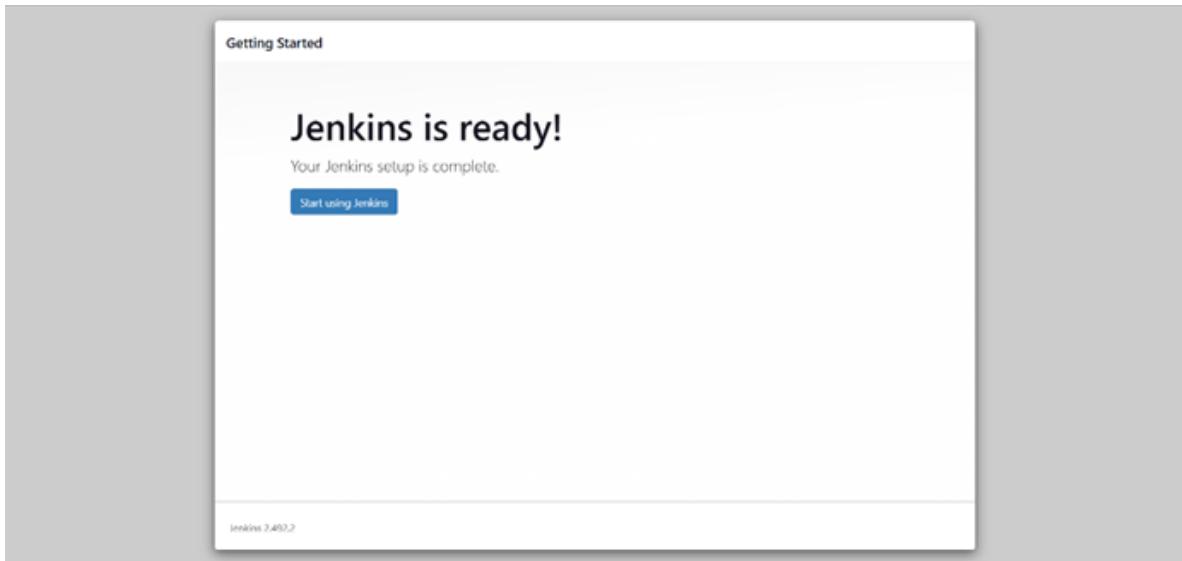
http://<VM_IP>

You should see the default Nginx welcome page.

Conclusion

You have successfully set up a Jenkins Freestyle Job to install Nginx on a local Ubuntu VM. This guide covers everything from Jenkins installation, configuration, and running the job to verify that Nginx is installed and running correctly.

Now, your Jenkins automation is ready to deploy Nginx effortlessly! 🎉



The screenshot shows the Jenkins user interface. The title bar indicates the URL is 'localhost:8080/user/admin/'. The main header says 'Jenkins' with a user icon and the name 'Yugeshwaran'. The top navigation bar includes links for Firewall Authentication Keep, Document from Yugeshwara, ACFrOgBPXoSQOqAUyRnO, New repository, Jugeshwaran [Jenkins], Jenkins Password Recovery, a search icon, a warning icon, the user name 'Yugeshwaran', and a 'log out' link. The left sidebar has a 'Status' tab (which is active), 'Builds', 'My Views', 'Account' (which is also active), 'Appearance', 'Preferences', 'Security', 'Experiments', and 'Credentials'. The right side of the screen shows the 'Yugeshwaran' account settings, including the Jenkins User ID: admin. At the bottom, it shows the URL 'localhost:8080/user/admin/account/' and links for 'REST API' and 'Jenkins 2.492.2'.

The screenshot shows the Jenkins dashboard. At the top, there's a navigation bar with the Jenkins logo, a search icon, a shield icon with a red dot, a user icon labeled 'admin', and a 'log out' link. Below the bar, a 'Dashboard' link is visible. On the left, there are links for 'New Item', 'Build History', 'Manage Jenkins', and 'My Views'. A 'Build Queue' section shows 'No builds in the queue.' A 'Build Executor Status' section shows '0/2'. In the center, a table displays a single build item: 'Install-Nginx'. The table has columns for Status (green checkmark), Working (yellow sun icon), Name (Install-Nginx), Last Success (3 min 40 sec ago, #2), Last Failure (N/A), and Last Duration (15 sec). Below the table, there are icons for 'S' (Stable), 'M' (Medium), and 'L' (Large). At the bottom right, there are links for 'REST API' and 'Jenkins 2.492.2'.

