

DevOps Training-Day-1

Installing Ubuntu on VirtualBox

Step 1: Download Required Files

1. **Download VirtualBox** from the official website:
👉 <https://www.virtualbox.org/>
2. **Download Ubuntu ISO** from the official Ubuntu website:
👉 <https://ubuntu.com/download/desktop>

Step 2: Install VirtualBox

1. Open the downloaded **VirtualBox** installer and follow the on-screen instructions to install it.
2. Once installed, launch **VirtualBox**.

Step 3: Create a New Virtual Machine

1. Click "**New**" in VirtualBox.
2. Enter a name (e.g., "Ubuntu VM").
3. Choose **Type: Linux**
4. Choose **Version: Ubuntu (64-bit)**
5. Click **Next**.

Step 4: Allocate Memory (RAM)

- Assign at least **2GB (4096 MB)** RAM (Recommended: 4GB or more).
- Click **Next**.

Step 5: Create a Virtual Hard Disk

1. Choose "**Create a virtual hard disk now**" → Click **Create**.
2. Select **VDI (VirtualBox Disk Image)** → Click **Next**.
3. Select **Dynamically allocated** → Click **Next**.
4. Set at least **25GB** storage (Recommended: 50GB or more).
5. Click **Create**.

Step 6: Attach Ubuntu ISO

1. Select the created VM from the list.
2. Click **Settings** → **Storage**.
3. Under **Controller: IDE**, click **Empty**.
4. Click the **CD icon** on the right → **Choose a disk file**.
5. Select the downloaded **Ubuntu ISO** file.
6. Click **OK**.

Step 7: Start the Virtual Machine

1. Select your **Ubuntu VM** → Click **Start**.
2. Ubuntu installer will launch.

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/debianstable 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:

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`

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.

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

[Continue](#)

Getting Started

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding	<div><div>** Plain Credentials</div><div>** Variant</div><div>** SSH Credentials</div><div>Credentials Binding</div><div>** SCM API</div><div>** Pipeline: API</div><div>** commons-lang3 v3.x Jenkins API</div><div>Timestamper</div><div>** Caffeine API</div><div>** Script Security</div><div>** JavaBeans Activation Framework (JAF) API</div><div>** JAXB</div><div>** SnakeYAML API</div><div>** JSON Api</div><div>** Jackson 2 API</div><div>** commons-text API</div><div>** Pipeline: Supporting APIs</div><div>** Plugin Utilities API</div><div>** Font Awesome API</div><div>** - required dependency</div></div>
✓ Timestamper	🔄 Workspace Cleanup	🕒 Ant	🔄 Gradle	
🔄 Pipeline	🔄 GitHub Branch Source	🔄 Pipeline: GitHub Groovy Libraries	🔄 Pipeline Graph View	
🕒 Git	🕒 SSH Build Agents	🔄 Matrix Authorization Strategy	🕒 PAM Authentication	
🔄 LDAP	🔄 Email Extension	🕒 Mailer	🔄 Dark Theme	

Jenkins is ready!

You have skipped the configuration of the Jenkins URL.

To configure the Jenkins URL, go to "Manage Jenkins" page.

Your Jenkins setup is complete.

Start using Jenkins


Create First Admin User

Username

Password

Confirm password

Full name

Jenkins

🔍

🔔 1

🛡️ 1

👤 mohana ▾


🚪 log out


Dashboard > All > New Item


New Item


Enter an item name

Select an item type


 **Freestyle project**
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

 **Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

 **Multi-configuration project**
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

 **Folder**

OK

Jenkins

🔍

🔔 1

🛡️ 1


👤 mohana ▾


🚪 log out


Dashboard >

+

New Item

 Build History

 Manage Jenkins

 My Views

Build Queue ▾
No builds in the queue.

Build Executor Status 0/2 ▾


Welcome to Jenkins!


This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.


Start building your software project

Create a job +

Set up a distributed build

Set up an agent 

Configure a cloud 

Learn more about distributed builds 

REST API

Jenkins 2.492.2

Configure

- General
- Source Code Management
- Triggers
- Environment**
- Build Steps
- Post-build Actions

Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

☐ Delete workspace before build starts

☐ Use secret text(s) or file(s) ?

Filter

Execute Windows batch command

Execute shell

Invoke Ant

Invoke Gradle script

Invoke top-level Maven targets

Run with timeout

Set build status to "pending" on GitHub commit

Add build step ^

code compilation, testing, and deployment.

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

Add post-build action ^

Save

Apply

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.