ChatGPT

Multipass VM Setup Guide for Serial Fixture Project

This guide walks you through the full process of:

- Installing a Linux VM using Multipass
- Cloning a private GitHub repository (with a Personal Access Token)
- Setting up Docker and Docker Compose
- Running your serial fixture project in a clean, Linux-native environment

---

Prerequisites (on macOS)

Install Homebrew if you haven't:

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

Then install Multipass:

```
brew install --cask multipass
```

---

1. Launch and Access the VM

```
multipass launch --name serial-test --mem 2G --disk 10G
multipass shell serial-test
```

---

2. Clone Private GitHub Repo with Personal Access Token

⚠ Replace `<your_token>` with your real GitHub **Personal Access Token (PAT)** and `<username>` with your GitHub username.

```
git clone https://<username>:<your_token>@github.com/<username>/<repo>.git
cd <repo>
```

Example:

```
git clone https://TEZZERAKTmedia:github_pat_abc123YOURTOKEN@github.com/
TEZZERAKTmedia/serial-fixture-devops-demo.git
cd serial-fixture-devops-demo
```

You can optionally store your credentials so you don't have to re-enter the token:

```
git config --global credential.helper store
```

---

3. Install Docker + Compose Plugin

```
sudo apt update
sudo apt install -y docker.io docker-compose-plugin

# Allow your user to run Docker without sudo:
sudo usermod -aG docker $USER
newgrp docker
```

Check versions:

```
docker --version
docker compose version
```

You should see something like:

```
Docker version 24.x.x, build ...
Docker Compose version v2.x.x
```

---

4. Build and Run the Project

Inside your cloned project folder:

```
docker compose up --build
```

You should see logs from both the test fixture and the simulated DUT.

Bonus: Quick Access to the VM

Add an alias to your shell config (optional):

```
echo "alias serial-vm='multipass shell serial-test'" >> ~/.zshrc
source ~/.zshrc
```

Now just run:

```
serial-vm
```

To jump into your Linux testing environment instantly.

Summary

You now have:

- A fully isolated Linux test environment
- Docker + Compose installed
- Your project cloned and running with `docker compose`
- GitHub access via token

This is ideal for working around macOS Docker limitations and simulating real embedded Linux deployments.

Let me know if you'd like a script version of this as a `.sh` file!