#!/bin/bash

# Function to run a shell command and handle errors

run\_command() {

echo -e "\n$2"

$1

if [ $? -ne 0 ]; then

echo "Error: $2"

exit 1

fi

}

# Step 1: Install Dependencies

install\_dependencies() {

echo "Step 1: Installing dependencies..."

run\_command "sudo apt update && sudo apt upgrade -y" "Updating package lists and upgrading system..."

run\_command "sudo apt install cmake gcc g++ git build-essential libpng-dev python3-pip wget curl -y" "Installing build tools and dependencies..."

}

# Step 2: Install Box64

setup\_box64() {

echo "Step 2: Setting up Box64..."

run\_command "git clone https://github.com/ptitSeb/box64.git" "Cloning Box64 repository..."

run\_command "cd box64 && mkdir build && cd build && cmake .. -DCMAKE\_BUILD\_TYPE=RelWithDebInfo && make" "Building Box64 from source..."

run\_command "cd box64/build && sudo make install" "Installing Box64..."

run\_command "sudo ldconfig" "Refreshing library paths for Box64..."

}

# Step 3: Install Box86

setup\_box86() {

echo "Step 3: Setting up Box86..."

run\_command "git clone https://github.com/ptitSeb/box86.git" "Cloning Box86 repository..."

run\_command "cd box86 && mkdir build && cd build && cmake .. -DCMAKE\_BUILD\_TYPE=RelWithDebInfo && make" "Building Box86 from source..."

run\_command "cd box86/build && sudo make install" "Installing Box86..."

run\_command "sudo ldconfig" "Refreshing library paths for Box86..."

}

# Step 4: Install Wine

setup\_wine() {

echo "Step 4: Installing Wine..."

run\_command "sudo dpkg --add-architecture i386 && sudo mkdir -pm755 /etc/apt/keyrings" "Setting up i386 architecture for Wine..."

run\_command "sudo wget -O /etc/apt/keyrings/winehq-archive.key https://dl.winehq.org/wine-builds/winehq.key" "Downloading WineHQ key..."

run\_command "sudo wget -NP /etc/apt/sources.list.d/ https://dl.winehq.org/wine-builds/ubuntu/dists/$(lsb\_release -cs)/winehq-$(lsb\_release -cs).sources" "Setting up Wine repository..."

run\_command "sudo apt update && sudo apt install --install-recommends winehq-stable -y" "Installing Wine Stable version..."

}

# Step 5: Install DXVK

setup\_dxvk() {

echo "Step 5: Installing DXVK..."

run\_command "wget https://github.com/doitsujin/dxvk/releases/download/v2.3/dxvk-2.3.tar.gz" "Downloading DXVK package..."

run\_command "tar -xvf dxvk-2.3.tar.gz" "Extracting DXVK package..."

run\_command "cd dxvk-2.3 && sudo ./setup\_dxvk.sh install" "Installing DXVK..."

}

# Step 6: Install Steam

setup\_steam() {

echo "Step 6: Installing Steam for 64-bit architecture..."

run\_command "sudo add-apt-repository multiverse -y" "Enabling multiverse repository..."

run\_command "sudo apt update" "Updating package list after adding multiverse repository..."

run\_command "sudo apt install steam -y" "Installing Steam..."

}

# Function to check if Steam is installed

check\_steam\_installed() {

if ! command -v steam > /dev/null; then

echo "Steam is not installed. Please install Steam first using your package manager or Box64."

exit 1

fi

}

# Function to run Steam via Box64

run\_steam() {

echo "Running Steam using Box64..."

# Set the environment for Box64, Wine, and DXVK

export BOX86\_PATH=/usr/local/bin/box86

export BOX64\_PATH=/usr/local/bin/box64

export WINEPREFIX=$HOME/.wine

export DXVK\_LOG\_LEVEL=none

export WINEARCH=win64

# Ensure the necessary libraries for Wine and DXVK are properly linked

sudo ldconfig

# Set up DXVK and Vulkan environment variables (ensure Vulkan drivers are properly installed)

export VK\_ICD\_FILENAMES=/usr/share/vulkan/icd.d/nvidia\_icd.json # Adjust path based on Vulkan drivers

export DXVK\_HUD=1 # Optional: enable DXVK HUD for diagnostics

# Launch Steam through Box64

$BOX64\_PATH /usr/games/steam

}

# Main function

main() {

install\_dependencies

setup\_box64

setup\_box86

setup\_wine

setup\_dxvk

setup\_steam

echo "All installations are complete!"

# After installation, run Steam

check\_steam\_installed

run\_steam

}

main