How to put OctoPrint on Linux?

Change everything in **RED** to your username.

Anything in code boxes you can just copy and paste them into the terminal.

For this tutorial, I will be using Ubuntu server 18.04. But you can use normal Ubuntu as well.

Putting Ubuntu on the computer:

First, we will need to download Ubuntu server. The for putting Ubuntu on the computer, I recommend downloading Rufus as the install media. Once you have put Ubuntu on a flash drive insert it into a USB slot and boot from the USB by going into BIOS or UEFI and selecting the flash drive. Then follow the prompts on the screen.

SSH into Ubuntu:

To SSH into Ubuntu you will need a program called Putty. Once you have downloaded Putty you will need to get the IP address of your Ubuntu computer. To get the IP address of your computer you will need to type the command ifconfig -a. Then look for the enp#s or inet then you should see an IP address that looks something like this (192.168.123.456). Then put that into Putty's Host name(or IP address). Lastly, click open and click "yes" then the terminal should open.

Updating Ubuntu:

For updating Ubuntu, you will need to do just two commands. sudo apt update and after that one is done and type the command sudo apt upgrade. Lastly, after that, you will need to restart your Ubuntu computer with the command sudo reboot now.

Note: I would set a static IP address to the Ubuntu computer, so you don't have to get the IP every time it reboots.

Installing OctoPrint/Testing:

1. We will need to make sure we are in our home directory. Command: cd ~
2. Then we will need to install the necessary python libraries and files. Command: sudo apt install python-pip python-dev python-setuptools python-virtualenv git libyaml-dev build-essential
3. Next, we need to make a directory to put OctoPrint in and then go inside the directory. Command: mkdir OctoPrint && cd OctoPrint
4. Now we are going to get ready to visualize OctoPrint. Command: virtualenv venv
5. Now we must active venv. Command: source venv/bin/activate
6. Install pip/ updating it. Command: pip install pip –upgrade
7. Installing Octoprint using pip. Command: pip install https://get.octoprint.org/latest
8. Giving permissions to the user account. Command: sudo usermod -a -G tty **username**

9. Giving more permissions to the user account. Command: sudo usermod -a -G dialout **username**
10. Setting OctoPrint to serve in venv. Command: ~/OctoPrint/venv/bin/octoprint serve
11. Testing OctoPrint. To test OctoPrint you will need to go to your web browser and type the IP address of your Ubuntu computer and add the port "5000" to the end of it to see if it works.

Making OctoPrint Start up after reboot:

1. Getting a file from GitHub. Command: wget https://github.com/foosel/OctoPrint/raw/master/scripts/octoprint.init && sudo mv octoprint.init /etc/init.d/octoprint
2. Getting more files from GitHub. Command: wget https://github.com/foosel/OctoPrint/raw/master/scripts/octoprint.default && sudo mv octoprint.default /etc/default/octoprint
3. Lastly, we will need to make the file executable. Command: sudo chmod +x /etc/init.d/octoprint

Editing the user and removing comments in the default file:

1. First, enter the file in a text editor. Command: sudo nano /etc/default/octoprint
    1. Now we must change some things in the file. Example:

 # Configuration for /etc/init.d/octoprint

 # The init.d script will only run if this variable non-empty.
*OCTOPRINT_USER=**Username***

 # base directory to use
*BASEDIR=/home/**Username**/.octoprint*

 # configuration file to use
*CONFIGFILE=/home/**Username**/.octoprint/config.yaml*

 # On what port to run daemon, default is 5000
PORT=5000

 # Path to the OctoPrint executable, you need to set this to match your installation!
*DAEMON=/home/**Username**/OctoPrint/venv/bin/octoprint*

 # What arguments to pass to octoprint, usually no need to touch this
DAEMON_ARGS="--port=$PORT"

 # Umask of files octoprint generates, Change this to 000 if running octoprint as its own, separate user
UMASK=022

 # Process priority, 0 here will result in a priority 20 process.
 # -2 ensures Octoprint has a slight priority over user processes.
NICELEVEL=-2

 # Should we run at startup?
START=yes

Adding default to AutoStart:

We will need to add default to run in AutoStart. Command: sudo update-rc.d octoprint defaults

Checking if OctoPrint is Active:

We need to see if OctoPrint is Working Right. Command: sudo service octoprint status. If you see text that says **Active in green,** then you are good so far.

Add users to sudoers file so it can run shutdown commands in OctoPrint:

We need to add the users to the sudoers. Because if you want to be able to shut down or reboot the OctoPrint server or Ubuntu computer in the OctoPrint GUI you will need to do this. Command: sudo nano /etc/sudoers.d/octoprint-shutdown. Then when you're in that file type "***Username*** *ALL=NOPASSWD: /sbin/shutdown*"

Installing haproxy:

1. We are going to use haproxy to remove the port 5000. You don't have to type it every time. Command: sudo apt install haproxy
2. Next, we are going to remove the old config file and make our own. But save the old one. Command: sudo cp /etc/haproxy/haproxy.cfg /etc/haproxy/haproxy.cfg_old Then do this Command: sudo rm /etc/haproxy/haproxy.cfg
3. Now we are going to make a new config file and edit the config file. Command: sudo nano /etc/haproxy/haproxy.cfg Then when you're in the file it should be empty. Paste this:

global
maxconn 4096
user haproxy
group haproxy
daemon
log 127.0.0.1 local0 debug

defaults
log global
mode http
option httplog
option dontlognull
retries 3

```
    option redispatch
    option http-server-close
    option forwardfor
    maxconn 2000
    timeout connect 5s
    timeout client 15min
    timeout server 15min

frontend public
    bind :::80 v4v6
    use_backend webcam if { path_beg /webcam/ }
    default_backend octoprint

backend octoprint
    reqrep ^([^\ :]*)\ /(.*)    \1\ /\2
    option forwardfor
    server octoprint1 127.0.0.1:5000

backend webcam
    reqrep ^([^\ :]*)\ /webcam/(.*)    \1\ /\2
    server webcam1 127.0.0.1:8080


Enable haproxy:
```

4. Then we are going to edit the haproxy default file. Command: sudo nano /etc/default/haproxy After you are in the file change the file to this.

```
# Defaults file for HAProxy
#
# This is sourced by both, the initscript and the systemd unit file, so do not
# treat it as a shell script fragment.


# Change the config file location if needed
#CONFIG="/etc/haproxy/haproxy.cfg"


# Add extra flags here, see haproxy(1) for a few options
#EXTRAOPTS="-de -m 16"
ENABLE=1
```

5. Next, we are going to check to see if haproxy is work right. You should see **green** text that says **Active**. Command: sudo service haproxy status

6. Lastly, we need to restart haproxy to save any changes we have made. Command: sudo service haproxy restart

Installing Webcam Support:

If you use OctoPrint right now you might not have Webcam support. Let's fix that.

1. First, we need to go to the home directory. Command: cd ~
2. Now we need to install the files needed. Command: sudo apt install subversion libjpeg8-dev imagemagick ffmpeg libv4l-dev cmake
3. Getting more files that are needed. Command: git clone https://github.com/jacksonliam/mjpg-streamer.git
4. Moving directory to mjpg-streamer-experimental. Command: cd mjpg-streamer/mjpg-streamer-experimental
5. Exporting libraries. Command: export LD_LIBRARY_PATH=.
6. Then make. Command: makei "./i
7. Test mjpg steamer. Command: sudo ./mjpg_streamer -nput_uvc.so" -o "./output_http.so"

Creating a Webcam startup script:

Side note do not use "sudo" in these commands.

1. First, we need to go to the home directory. Command: cd ~
2. Making a directory or our scripts. Command: mkdir scripts
3. Then we are going to create a file in the script's directory. Then edit that file.
   Command: nano /home/**Username**/scripts/webcam

After you are in the webcam file paste this in it.

```
#!/bin/bash
# Start / stop streamer daemon


case "$1" in
  start)
    /home/chris/scripts/webcamDaemon >/dev/null 2>&1 &
    echo "$0: started"
    ;;
  stop)
```

```
        pkill -x webcamDaemon
        pkill -x mjpg_streamer
        echo "$0: stopped"
        ;;
    *)
        echo "Usage: $0 {start|stop}" >&2
        ;;
esac
```

Creating a Daemon scripts:

Side note do not use "sudo" in these commands.

1. We need to make a Daemon script, so the webcam works right.

   Command: nano /home/**Username**/scripts/webcamDaemon

After you are in the file pasta this in it.

```
#!/bin/bash

MJPGSTREAMER_HOME=/home/chris/mjpg-streamer/mjpg-streamer-experimental
MJPGSTREAMER_INPUT_USB="input_uvc.so"
MJPGSTREAMER_INPUT_RASPICAM="input_raspicam.so"

# init configuration
camera="auto"
camera_usb_options="-r 640x480 -f 10"
camera_raspi_options="-fps 10"

if [ -e "/boot/octopi.txt" ]; then
   source "/boot/octopi.txt"
fi

# runs MJPG Streamer, using the provided input plugin + configuration
function runMjpgStreamer {
   input=$1
   pushd $MJPGSTREAMER_HOME
   echo Running ./mjpg_streamer -o "output_http.so -w ./www" -i "$input"
   LD_LIBRARY_PATH=. ./mjpg_streamer -o "output_http.so -w ./www" -i "$input"
   popd
}
```

```
# starts up the RasPiCam
function startRaspi {
   logger "Starting Raspberry Pi camera"
   runMjpgStreamer "$MJPGSTREAMER_INPUT_RASPICAM $camera_raspi_options"
}

# starts up the USB webcam
function startUsb {
   logger "Starting USB webcam"
   runMjpgStreamer "$MJPGSTREAMER_INPUT_USB $camera_usb_options"
}

# we need this to prevent the later calls to vcgencmd from blocking
# I have no idea why, but that's how it is...
vcgencmd version

# echo configuration
echo camera: $camera
echo usb options: $camera_usb_options
echo raspi options: $camera_raspi_options

# keep mjpg streamer running if some camera is attached
while true; do
   if [ -e "/dev/video0" ] && { [ "$camera" = "auto" ] || [ "$camera" = "usb" ] ; }; then
      startUsb
   elif [ "`vcgencmd get_camera`" = "supported=1 detected=1" ] && { [ "$camera" = "auto" ] ||
[ "$camera" = "raspi" ] ; }; then
      startRaspi
   fi

   sleep 120
done
```

Edit Webcam permissions:

We need to edit permissions to some of the webcam files so they can run in startup. Command: **sudo chmod +x /home/Username/scripts/webcam** Then do this command **sudo chmod +x /home/Username/scripts/webcamDaemon**

Add webcam(‘s) to startup:

Adding webcam to startup so you don't have to manually start it every time you reboot your computer. Command: **sudo nano /etc/rc.local**

After you are in the file change it to this.

```
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "" on success or any other
# value on error.
#
# To enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

/home/Username/scripts/webcam start

exit 0
```

Start Webcam:

Now let's start the webcam. Command: **sudo /home/Username/scripts/webcam start**

Change more permissions:

Changing permissions. Command: **sudo chmod +x /etc/rc.local**

Installing Avahi:

Avahi is like a DNS server. You must type the IP address of your computer every time you want to get into OctoPrint.

1. Installing Avahi. Command: **sudo apt install avahi-daemon**
2. Edit the host-name file. **sudo apt install avahi-daemon** *You can name this whatever you want to.*
3. Edit Hosts file. Command: **sudo nano /etc/hosts**

After you are in the file pasta this in it.

```
127.0.0.1       localhost.localdomain   localhost
::1             localhost6.localdomain6 localhost6

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff02::1 ip6-allnodes
```

ff02::2 ip6-allrouters
ff02::3 ip6-allhosts

127.0.1.1 example host-name
↑
Whatever you named the host name

Lastly Reboot:

Now that everything is done. It's good to do a reboot of your Ubuntu Computer to save and make sure everything is work.

Command: **sudo reboot now**

Helpful Notes:

If you want OctoPrint GUI shutdown and reboot etc buttons to work you will need these commands.

- Restart OctoPrint: sudo service octoprint restart
- Restart system: sudo shutdown -r now
- Shutdown system: sudo shutdown -h now

Webcam links:

- Stream URL: /webcam/?action=stream
- Snapshot URL: http://127.0.0.1:8080/?action=snapshot
- Path to FFMPEG: /usr/bin/ffmpeg

Now put this links/Commands in OctoPrint settings in their designated areas.