

**First of all we need to make 100% sure that the system is up to date. SSH into your rpi and type the following commands:**

```
sudo apt update
```

```
sudo apt full-upgrade
```

### **Klipper Installation**

Follow the instructions to install Klipper if you havent already.

- klipper: <https://www.klipper3d.org/Installation.html>

### **Installation Requirements**

Start by making sure that the following configuration options are written in your printer.cfg file:

```
[virtual_sdcard]
```

```
path:
```

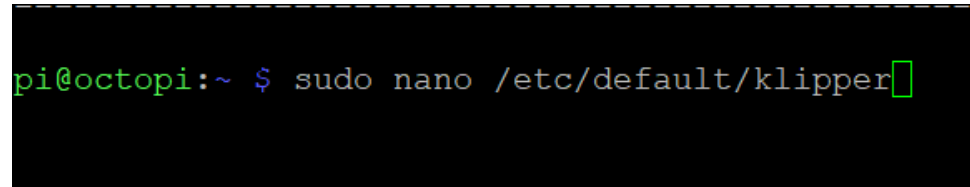
```
~/octoprint/uploads/
```

```
[pause_resume]
```

```
[display_status]
```

**SSH into your rpi with something like putty. In the terminal window type/copy and paste the following:**

```
sudo nano /etc/default/klipper
```



```
pi@octopi:~ $ sudo nano /etc/default/klipper
```

Change the script so it looks like the following. This points klipper to the printer\_config directory that you are about to make.

*# Configuration for /etc/init.d/klipper*

```
KLIPPY_USER=pi KLIPPY_EXEC=/home/pi/klippy-env/bin/python
KLIPPY_ARGS="/home/pi/klipper/klippy/klippy.py /home/pi/printer_config/printer.cfg -l /tmp/klippy.log
-a /tmp/klippy_uds"
```



```
GNU nano 3.2 /etc/default/klipper
# Configuration for /etc/init.d/klipper
KLIPPY_USER=pi
KLIPPY_EXEC=/home/pi/klippy-env/bin/python
KLIPPY_ARGS="/home/pi/klipper/klippy/klippy.py /home/pi/printer_config/printer.cfg -l /tmp/klippy.log -a /tmp/klippy_uds"
```

Once you have done this press Ctrl + x to prompt exit file and then press y to save the changes.

Now create a folder called printer\_config in /home/pi and move your printer.cfg file to the directory, then move into the newly created printer\_config directory:

```
cd /home/pi
mkdir printer_config
mv printer.cfg printer_config
cd printer_config
```

Now create a file called moonraker.conf in this directory:

```
sudo nano /home/pi/printer_config/moonraker.conf
```

Copy this text to the file for your basic moonraker config:

# Sample Moonraker Configuration File

[server]

# Bind server defaults of 0.0.0.0, port 7125

enable\_debug\_logging: True

[file\_manager]

config\_path: /home/pi/printer\_config

[database]

database\_path: ~/.moonraker\_database

[authorization]

enabled: True

trusted\_clients:

127.0.0.1

force\_logins: False

# Enter your client IP here or range here

cors\_domains:

# Allow CORS requests for Fluidt

<http://app.fluidt.xyz>

# Enable Octoprint compatibility for Slicer uploads Supports Cura,

# Slic3r, and Slic3r derivatives (PrusaSlicer, SuperSlicer)

[octoprint\_compat]

[update\_manager KlipperScreen]

type: git\_repo

path: ~/KlipperScreen

origin: <https://github.com/jordanruthe/KlipperScreen.git>

env: ~/.KlipperScreen-env/bin/python

requirements: scripts/KlipperScreen-requirements.txt

install\_script: scripts/KlipperScreen-install.sh

**If you need extra config for moonraker then you can read the documentation here:**

<https://moonraker.readthedocs.io/en/latest/configuration/>

**Now we can clone the moonraker git repository with these commands:**

cd

git clone <https://github.com/Arksine/moonraker.git>

**Then run the install with the following command-** (I could only seem to get moonraker working properly by installing it in the printer\_config directory. In the /home/pi directory I would get a error message about permissions) :

cd ~/moonraker/scripts ./install-moonraker.sh -f -c /home/pi/printer\_config/moonraker.conf

**Now moonraker should be working.**

**Reboot the system for good measures:**

sudo reboot now

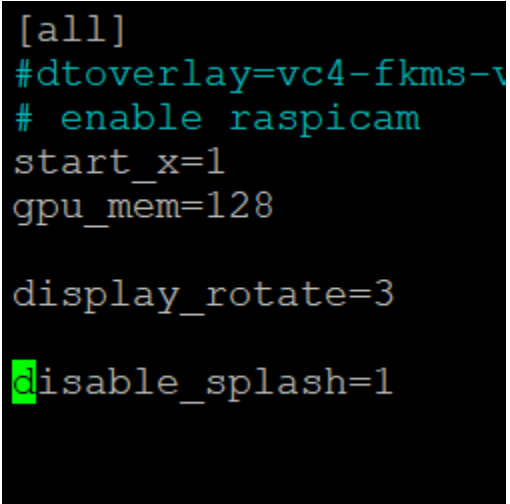
Now we can carry on and install klipperscreen with the following command:

```
cd ~/git clone https://github.com/jordanruthe/KlipperScreen.gitcd  
~/KlipperScreen./scripts/KlipperScreen-install.sh
```

Now we are going to modify some files. Type the following command:

```
sudo nano /boot/config.txt
```

Scroll to the bottom of the file and add these lines (only add display\_rotate if you are wanting to rotate your screen display. Options are from 0-4 from memory. You will need to reboot after the change to make sure it has rotated to where you want it)



```
[all]  
#dtoverlay=vc4-fkms-v  
# enable raspicam  
start_x=1  
gpu_mem=128  
  
display_rotate=3  
disable_splash=1
```

Press ctrl+x and then y to save.

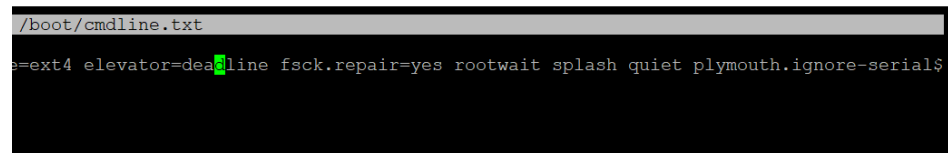
Then copy/paste this command:

```
sudo nano boot/cmdline.txt
```

At the end of the line after "rootwait" copy/paste the following:

```
splash quiet plymouth.ignore-serial-consoles logo.nologo vt.global_cursor_default=0
```

It should look something like this:



```
/boot/cmdline.txt
e=ext4 elevator=deadline fsck.repair=yes rootwait splash quiet plymouth.ignore-serial$
```

Now ensure that xinput and xserver and x11 is installed:

```
sudo apt-get install libx11-dev libxext-dev libxi-dev x11proto-input-dev
```

```
wget http://github.com/downloads/tias/xinput ... 7.5.tar.gz
```

```
sudo apt install xinput
```

Then

```
sudo apt install xserver-xorg-input-evdev
```

If you need to rotate the screen then run the following command to find the name of your screen:

```
DISPLAY=0 xinput --list
```

You should see something like this:

```
pi@octopi:~ $ DISPLAY=:0 xinput --list
❑ Virtual core pointer              id=2      [master pointer  (3)]
❑   Virtual core XTEST pointer      id=4      [slave pointer   (2)]
❑   STMicroelectronics KEDEI_touchscreem id=6      [slave pointer   (2)]
❑ Virtual core keyboard             id=3      [master keyboard (2)]
❑   Virtual core XTEST keyboard      id=5      [slave keyboard  (3)]
pi@octopi:~ $
```

(Notice that my screen has a typo in the name “STMicroelectronics KEDEI\_touchscreem” instead of “touchscreen”..... That held me up for quite some time lol.)

Now type the following but replace my screens name with the name of your device:

DISPLAY=:0 xinput --list-props “STMicroelectronics KEDEI touchscreen”

You should see something like this:

```
pi@octopi:~ $ DISPLAY=:0 xinput --list-props "STMicroelectronics KEDEI touchscreen"
Device "STMicroelectronics KEDEI touchscreen":
  Device Enabled (115):  1
  Coordinate Transformation Matrix (116):  0.000000, -1.000000, 1.000000, 1.000000, 0.000000, 0.000000, 0.000000, 0.000000, 1.000000
  Device Accel Profile (242):  0
  Device Accel Constant Deceleration (243):  1.000000
  Device Accel Adaptive Deceleration (244):  1.000000
  Device Accel Velocity Scaling (245):  10.000000
  Device Product ID (246):  3023, 5
  Device Node (247):  "/dev/input/event0"
  Evdev Axis Inversion (248):  0, 0
  Evdev Axis Calibration (249):  <no items>
  Evdev Axes Swap (250):  0
  Axis Labels (251):  "Abs MT Position X" (239), "Abs MT Position Y" (240), "Abs MT Pressure" (241), "None" (0), "None" (0), "None" (0)
  Button Labels (252):  "Button Unknown" (235), "Button Unknown" (235), "Button Unknown" (235), "Button Wheel Up" (121), "Button Wheel Down" (122)
  Evdev Scrolling Distance (253):  0, 0, 0
  Evdev Middle Button Emulation (254):  0
  Evdev Middle Button Timeout (255):  50
  Evdev Middle Button Button (256):  2
  Evdev Third Button Emulation (257):  0
  Evdev Third Button Emulation Timeout (258):  1000
  Evdev Third Button Emulation Button (259):  3
  Evdev Third Button Emulation Threshold (260):  20
  Evdev Wheel Emulation (261):  0
  Evdev Wheel Emulation Axes (262):  0, 0, 4, 5
  Evdev Wheel Emulation Inertia (263):  10
  Evdev Wheel Emulation Timeout (264):  200
  Evdev Wheel Emulation Button (265):  4
  Evdev Drag Lock Buttons (266):  0
pi@octopi:~ $
```

Take note of the factory coordinate transformation matrix. Now type this command to rotate the matrix (you may need a different matrix combination than mine depending on your screen. Just have a search around for combinations and you will find one eventually):

```
DISPLAY=0 xinput set-prop "STMicroelectronics KEDEI_touchscreen" 'Coordinate Transformation Matrix' 0 -1 1 1 0 0 0 1
```

Test the touch matrix to check that it is correct for the screen orientation.

Once satisfied then run the following command and add these lines:

```
sudo nano /etc/udev/rules.d/51-touchscreen.rules
```

Add the following line, changing your screen name and matrix to suit.

```
ACTION=="add", ATTRS{name}=="STMicroelectronics KEDEI_touchscreen",  
ENV{LIBINPUT_CALIBRATION_MATRIX}="0 -1 1 1 0 0 0 1 "
```

Press ctrl+x and then y to save.

Also modify this file and add this line:

```
sudo nano /etc/rc.local
```

```
do  
    echo " http://$name.local"  
done  
  
for ip in $(hostname -I);  
do  
    echo " http://$ip"  
done  
  
DISPLAY=:0 xinput set-prop "STMicroelectronics KEDEI_touchscreen" 'Coordinate Transformation Matrix' 0 -1 1 1 0 0 0 1
```



## **NOTE:**

### **Touchscreen Calibration**

Most people don't need to calibrate, but if you do need to calibrate your touchscreen, follow the below steps.

```
sudo add-apt-repository ppa:tias/xinput-calibrator-ppa
```

```
sudo apt-get install xinput_calibrator
```

Run this command:

```
DISPLAY=:0 xinput_calibrator --list
```

It will output something such as:

```
Device "wch.cn USB2IIC_CTP_CONTROL" id=6
```

Find the ID of your display and put it in the following command:

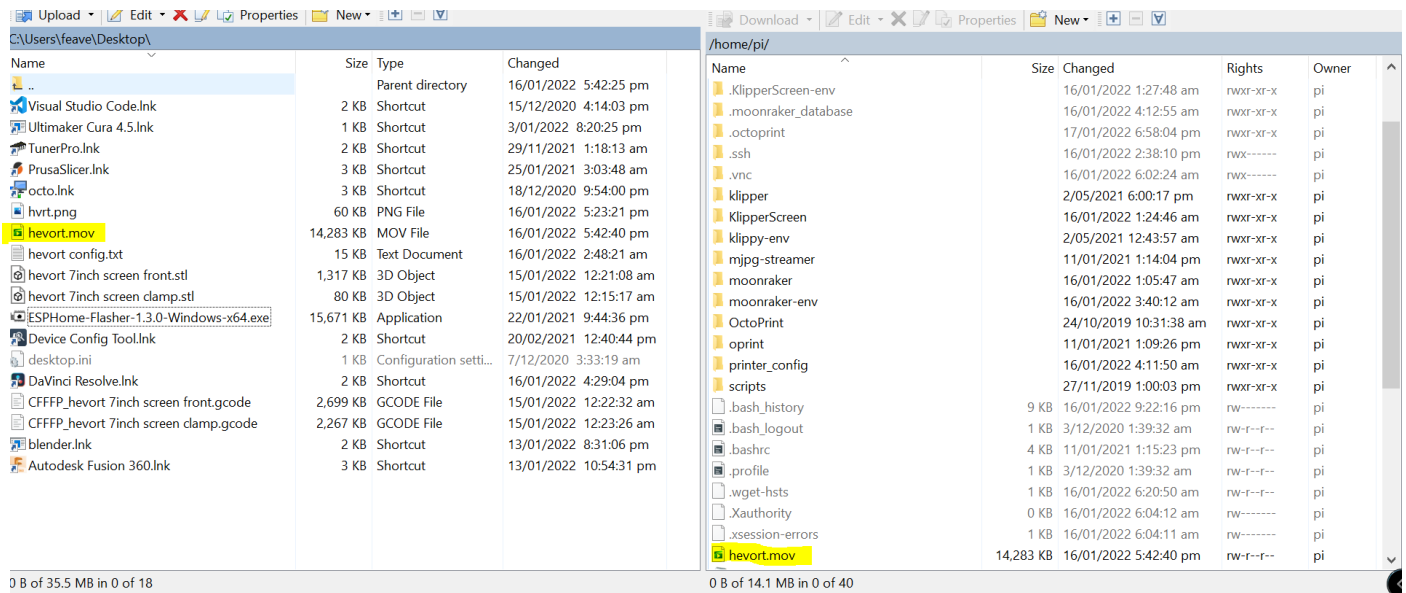
```
DISPLAY=:0 xinput_calibrator -v --device <id from last command>
```

### **Now install omxplayer:**

```
Sudo apt install omxplayer
```

**Now you can make your boot screen splash video. I used davinci resolve because to make mine because its free. There are plenty of tutorials on youtube.**

Next we need to transfer our video to the rpi. Im unsure how to do this via terminal so I used winscp to move the video file to /home/pi (named hevort.mov):



Now we can confirm the splash screen is working by running this command in the terminal:

```
omxplayer hevort.mov
```

All going well the splash screen should play. Now we can modify one more file to make the splash screen play on boot:

```
sudo nano /etc/rc.local
```

**Add this line at the bottom of the page before exit0:**

`omxplayer /home/pi/hevort.mov &`

```
echo
echo "https is also available, with a self-signed certificate."
echo
echo "-----"
echo

omxplayer /home/pi/hevort.mov &
exit 0
```

**Ctrl+x and y to save.**

**Congratulations! You should now have a working (maybe rotated) touchscreen with your own custom splash video!!!**



