

Sollte schon ein Raspi Betriebssystem auf der sD Karte vorhanden sein, und es unter Windows gesichert werden was mit Win32 disk imager nicht geht, so nimmt man

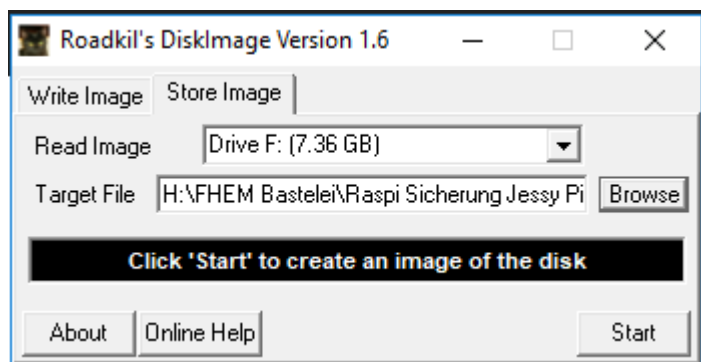
das hier:

<http://www.roadkil.net/program.php?ProgramID=12>

hier weitere Tips

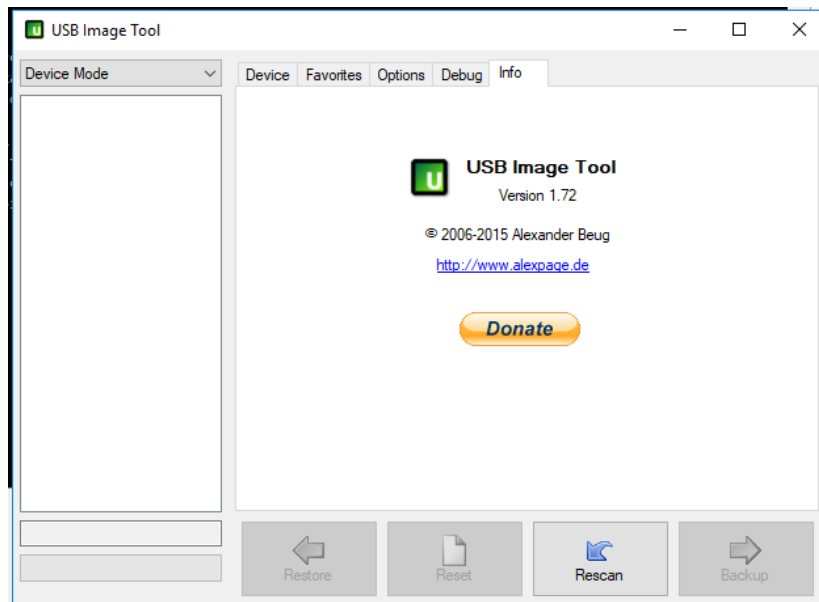
<https://linuxundich.de/raspberry-pi/linux-images-fuer-den-raspberry-pi-auf-sd-karte-installieren/>

beliebigen Verzeichnis entpacken. Danach ruft ihr die *DiskImage_1_6_WinAll.exe* am besten mit administrativen Rechten auf — Klickt dazu im Dateimanager mit der rechten Maustaste auf die EXE-Datei und wählt dann *als Administrator ausführen* aus. Im Programm geht ihr in den Reiter *Store Image*, wählt das zu sichernde Laufwerk aus und gebt noch den Speicherort der Zieldatei vor.



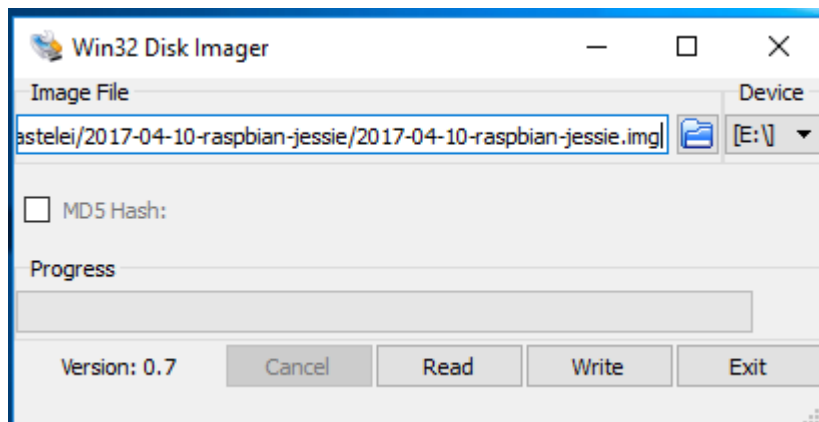
Mitdem Programm hates bei mir nicht geklappt da scheinbar der Bootsektor nicht kopiert wurde, also nach dem Sichern mal auf eine neue Karte kopieren und in den Raspi, wenn dann alles läuft ist gut

Alternative:



Mit Backup und restore hat alles geklappt und ich habe eine 2. SD Karte zum experimentieren.

Nach der Sicherung einer laufenden Installation kann man mit Roadkil,s auch eine neue Variante auf die SD Karte schreiben, alternativ natürlich auch mit Win32DiskImager der in den meisten Anleitungen empfohlen wird.

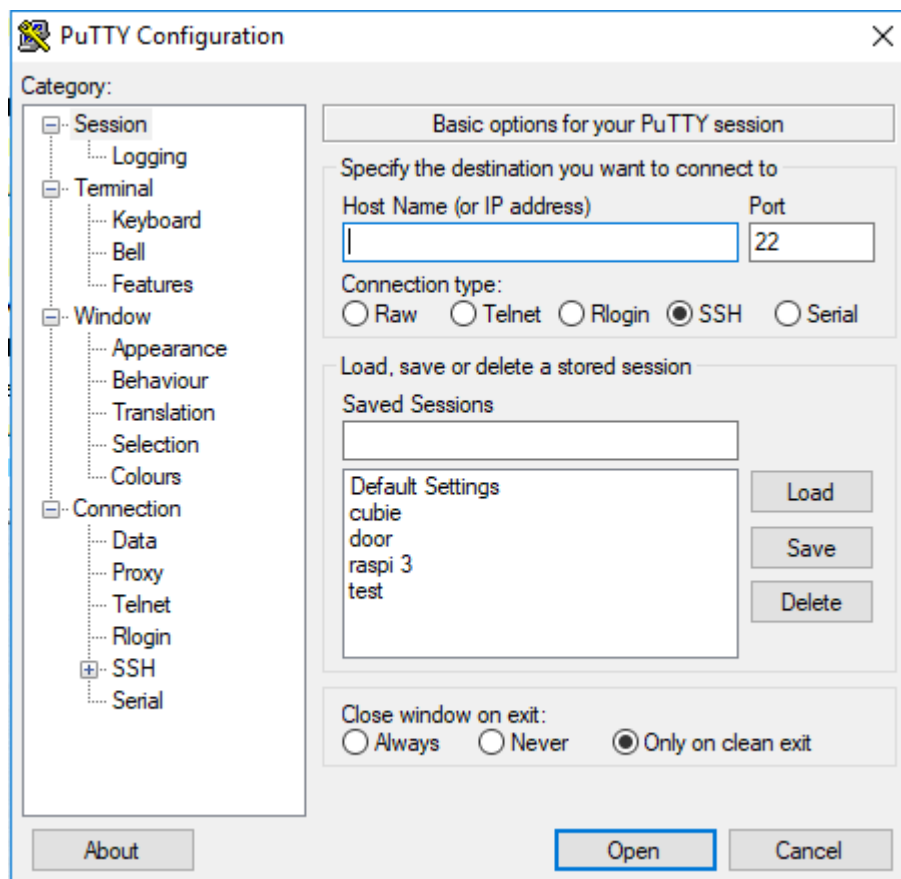


Als erstes die Speicherkarte mit dem raspi Image beschreiben
dann in den Raspi stecken und mit Netzwerkanbindung starten
also inFritzbox oder einem anderen Router die Netzwerkadresse ermitteln
Nun wird die IP-Adresse des RPI benötigt. Diese wird gewöhnlich vom Router (z.B. FritzBox)

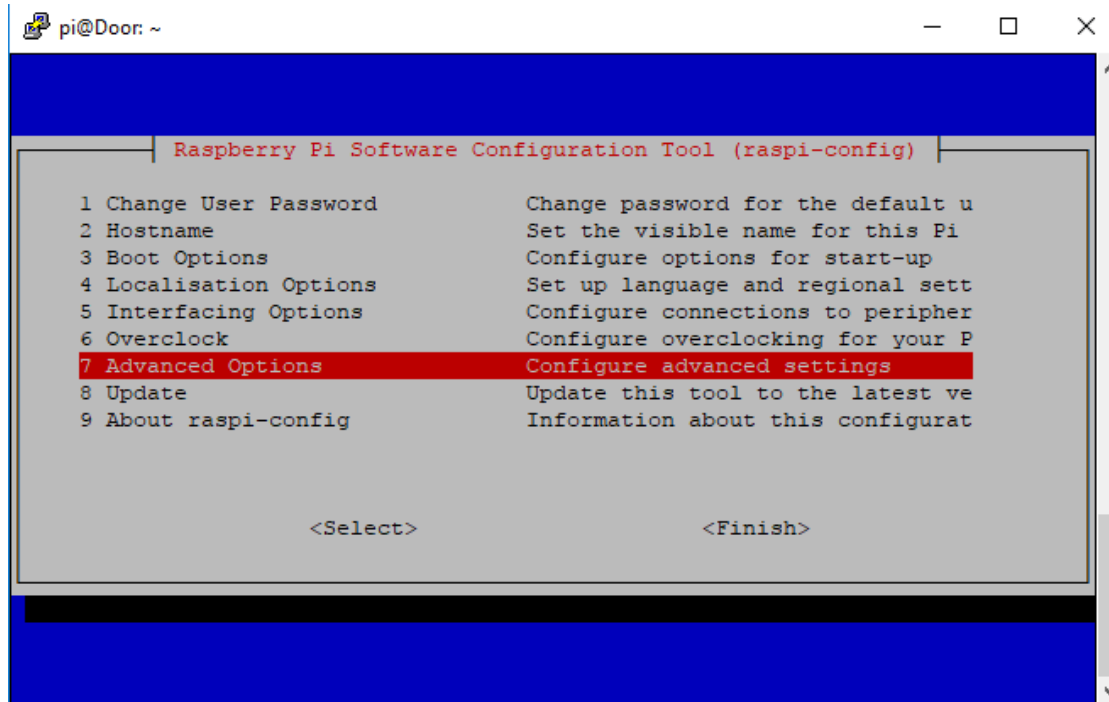
per DHCP zugewiesen und kann über das Router-Interface abgefragt werden. Bei einer FritzBox kann die Info im Menüpunkt Heimnetz -> Heimnetzübersicht eingesehen werden. Der RPI taucht hier gewöhnlich als "raspberrypi" auf.

In den Details wird dann noch der Haken bei "Diesem Netzwerkgerät immer die gleiche IPv4-Adresse zuweisen" gesetzt, damit sich die IP später nicht bei einem Neustart o.Ä. verändert.

Dann über putty mit dieser Ip auf den RPI zugreifen



Expand filesystem ! damit der gesamte Platz der SDKarte genutzt wird



danach noch Raspi-config mit der Option 8 auf die aktuellste Version updaten.

danach Reboot, die anderen Einstellungen wurden ja schon direkt am Anfang geändert.

7. SSH aktivieren und SSH-Schlüssel neu erstellen (optional)

Images von Distributionen enthalten einen Schlüssel für den SSH-Server, mit dem sich der Raspberry Pi gegenüber dem Client authentifiziert. Wenn man sich ein entsprechendes Image auf seine SD-Speicherkarte gezogen hat, dann ist dieser Schlüssel überall gleich. Ein Schlüssel sollte aber einzigartig sein, sonst eignet er sich nicht zur sicheren Authentifizierung. Deshalb sollte man bei einer Erstkonfiguration diesen Schlüssel ändern.

Zuerst löschen wir alle Dateien, in denen sich die Schlüssel befinden. Davon gibt es mehrere.

```
sudo rm /etc/ssh/ssh_host_*
```

Anschließend führen wir eine Rekonfiguration des SSH-Servers durch. Die Erstellung der Schlüssel-Dateien erfolgt automatisch.

```
sudo dpkg-reconfigure openssh-server
```

Bei einem neuen Raspbian-Image ist SSH standardmäßig deaktiviert bzw. abgeschaltet. Man kann es über "raspi-config" aktivieren oder auf der Kommandozeile aktivieren und starten.

Mittels Samba kann man z.B. den Ordner /opt/fhem als Share freigeben. Dieser Share kann

z.B. im Windows-Explorer als Laufwerk verbunden werden, so dass die Bearbeitung von config- und Programmdateien bequem möglich ist. Hier eine hilfreiche Kurzanleitung aus der (wenn man auf einen speziellen user verzichtet) nur die Einträge für smb.conf gesetzt werden müssen.

```
sudo apt-get install samba cifs-utils
```

Danach muss der share definiert werden mittels

<https://www.elektronik-kompodium.de/sites/raspberry-pi/2007071.htm>

```
sudo nano /etc/samba/smb.conf
```

Wird benötigt, um Mails versenden zu können, bspw. für Alarm-Benachrichtigungen.

Nach Installation des Paketes benötigt man noch eine Routine in FHEM gemäß E-Mail senden#Raspberry Pi

```
sudo apt-get install sendEmail
```

Wird z.B. für das Modul FRITZBOX benötigt, da es eine im Netzwerk vorhandene Fritzbox über deren Telnet-Port anspricht.

```
sudo apt-get install libnet-telnet-perl
```

Kann verwendet werden, um auf anderen Rechnern im Netzwerk Linux-Befehle oder Skripts auszuführen. Auch können auf Slave-FHEM-Installationen Befehl ausgeführt werden, z.B. mit

```
system("echo 'set lampe on' | /usr/bin/socat - TCP:1.2.3.4:7072");
```

1.2.3.4 muss natürlich durch die IP-Adresse des Zielrechners ersetzt werden.

```
sudo apt-get install socat
```

Perl libcrypt, erforderlich falls Homematic-devices mit AES verwendet werden sollen.

```
sudo apt-get install libcrypt-rijndael-perl
```

Installation nach fhem wiki zum Beispiel:

https://wiki.fhem.de/wiki/Raspberry_Pi

RPI in Betrieb nehmen und IP-Adresse ermitteln

```
sudo raspi-config ( falls Tastatur noch nicht umgestellt -liegt auf Altgr 3)
```

dann als erstes Tastatur etc umstellen,

Ländercode etc umstellen

dann immer das Passwort ändern.

aber erst nachdem die Tastatur richtig eingestellt wurde, sonst gibts mit z und y ev Probleme

pi

raspberry (Achtung wenn Feststelltaste aktiviert)

Runterfahren mit

sudo reboot

<https://forum.fhem.de/index.php/topic,15848.msg103268.html#msg103268>

Nimm einen Raspberry und installiere ein nacktes Raspbian OS.

Danach führst Du folgendes Skript aus: ev mit sudo vorweg

Code: [Auswählen]

```
rm /etc/rc.local
```

```
cd /opt
```

```
sudo apt-get -y update
```

```
sudo apt-get -y upgrade
```

```
wget http://fhem.de/fhem-5.8.tar.gz
```

```
tar xvf fhem-5.8.tar.gz
```

```
ln -s fhem-5.8 fhem
```

```
rm fhem-5.8.tar.gz
```

```
echo cd /opt/fhem >> /etc/rc.local
```

```
echo "perl fhem.pl fhem.cfg &" >> /etc/rc.local
```

```
echo exit 0 >> /etc/rc.local
```

```
chmod a+x /etc/rc.local
```

reboot

oder

Ein Post weiter

Oder noch einfacher:

Code: [Auswählen]

```
sudo apt-get update && apt-get -y upgrade
```

laden der aktuellen Version, ev die Zahlen ändern hier 5.8

```
sudo wget "http://fhem.de/fhem-5.8.deb"
```

entpacken der geladenen Dateien

```
sudo dpkg -i fhem-5.8.deb
```

entfernen der nicht mehr

```
sudo rm fhem-5.8.deb
```

Danach hast Du ein funktionsfähiges fhem.

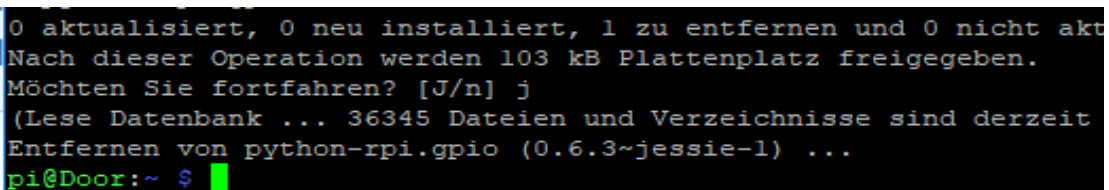
Doorpi installieren

<https://www.doorpi.org/forum/lexicon/entry/1-installation-doorpi-mittels-pypi-auf-einem-raspberry-pi-raspbian/>

Deinstallation von "unnötigen" #apt-get Paketen (kommen später über #pip / #easy_install wieder, wenn diese benötigt sind)

Shell-Script

```
sudo apt-get remove python-pip python-rpi.gpio
```



```
0 aktualisiert, 0 neu installiert, 1 zu entfernen und 0 nicht akt
Nach dieser Operation werden 103 kB Plattenplatz freigegeben.
Möchten Sie fortfahren? [J/n] j
(Lese Datenbank ... 36345 Dateien und Verzeichnisse sind derzeit
Entfernen von python-rpi.gpio (0.6.3~jessie-1) ...
pi@Door:~ $
```

Aktuelle notwendig ist die Installation von #python-dev per apt-get (siehe auch Bug bei der Installation von DoorPi)

Shell-Script

`sudo apt-get install python-dev`

```
Es müssen 18,3 MB an Archiven heruntergeladen werden.  
Nach dieser Operation werden 26,3 MB Plattenplatz zusätzlich benutzt.  
Möchten Sie fortfahren? [J/n] j
```

```
Trigger für man-db (2.7.0.2-5) werden verarbeitet ...  
libexpat1-dev:armhf (2.1.0-6+deb8u3) wird eingerichtet ...  
libpython2.7-dev:armhf (2.7.9-2+deb8ul) wird eingerichtet ...  
libpython-dev:armhf (2.7.9-1) wird eingerichtet ...  
python2.7-dev (2.7.9-2+deb8ul) wird eingerichtet ...  
python-dev (2.7.9-1) wird eingerichtet ...  
pi@Door:~ $
```

Betriebssystem auf dem RaspberryPi aktualisieren

Shell-Script

`sudo apt-get update && sudo apt-get -y upgrade && sudo apt-get -y dist-upgrade`

```
Paketlisten werden gelesen... Fertig  
Paketlisten werden gelesen... Fertig  
Abhängigkeitsbaum wird aufgebaut.  
Statusinformationen werden eingelesen.... Fertig  
Paketaktualisierung (Upgrade) wird berechnet... Fertig  
0 aktualisiert, 0 neu installiert, 0 zu entfernen und 0 nicht aktualisiert.  
Paketlisten werden gelesen... Fertig  
Abhängigkeitsbaum wird aufgebaut.  
Statusinformationen werden eingelesen.... Fertig  
Paketaktualisierung (Upgrade) wird berechnet... Fertig  
0 aktualisiert, 0 neu installiert, 0 zu entfernen und 0 nicht aktualisiert.  
pi@Door:~ $
```

Installation von pip über easy_install (Thema) sowie die Pakete #linphone4raspberrypi und #python-daemon, die in der requirements.txt sonst regelmäßig Probleme machen.

Shell-Script

`sudo easy_install pip || (wget https://bootstrap.pypa.io/ez_setup.py -O - | sudo python) && sudo easy_install pip`

`sudo pip install linphone4raspberrypi python-daemon`

```
creating /usr/local/lib/python2.7/dist-packages/pip-9.0.1-py2.7.egg  
Extracting pip-9.0.1-py2.7.egg to /usr/local/lib/python2.7/dist-packages  
Adding pip 9.0.1 to easy-install.pth file  
Installing pip script to /usr/local/bin  
Installing pip2.7 script to /usr/local/bin  
Installing pip2 script to /usr/local/bin  
  
Installed /usr/local/lib/python2.7/dist-packages/pip-9.0.1-py2.7.egg  
Processing dependencies for pip  
Finished processing dependencies for pip  
pi@Door:~ $
```



```
100% | 342KB 102KB/s
Collecting lockfile>=0.10 (from python-daemon)
  Downloading lockfile-0.12.2-py2.py3-none-any.whl
Installing collected packages: linphone4raspberry, docutils, lockfile, python-
emon
Successfully installed docutils-0.13.1 linphone4raspberry-3.9.0 lockfile-0.12.
python-daemon-2.1.2
pi@Door:~ $
```

Download, Installation und erster Start von DoorPi

Shell-Script

```
sudo pip install doorpi && sudo doorpi_cli --trace
```

Nun ist DoorPi installiert und das erste Mal gestartet. Die Weboberfläche (siehe FAQ) ist erreichbar und die passende URL wird von DoorPi ausgegeben:

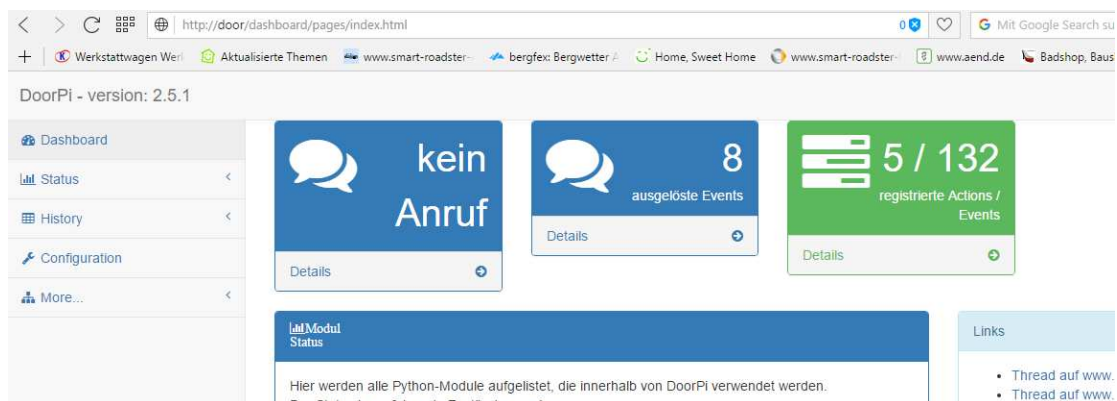
Eventuell kann auch ein #Port angegeben sein, wenn DoorPi nicht an Port 80 gebunden werden konnte (Default-Ausweichport wäre 8080).

Im Windows Explorer

door:8080 oder eben door:80 eingeben, door in meinem Fall Name des Raspis, in rasp-config definiert

Benutzer: door

Passwort pi



Mit der Tastenkombination [Strg] + [c] beende ich DoorPi und prüfe die entstandenen Dateien:

```
Datei /usr/local/bin/doorpi_cli
```

```
pi@Door:/etc/init.d $ cd /usr/local/bin/
```

```
pi@Door:/usr/local/bin $ ls -l
```

```
doorpi_cli
```

vorhanden!

Datei /etc/init.d/doorpi (falls nicht vorhanden -> Neuerstellung daemon-file)

```
pi@Door:~ $ cd /etc/init.d/
```

```
pi@Door:/etc/init.d $ ls -l
```

bei mir nicht vorhanden daher Neuerstellung (Link im Original Artikel)

und nun der Knackpunkt, nach 3 Fehlversuchen mit dem daemon file hab ich das hier gefunden:

<https://www.doorpi.org/forum/thread/12-doorpi-als-daemon-dienst-service-autostart-unter-jessie-einrichten/?pageNo=2>

wenn das Thema noch aktuell ist, hatte den auch diesen Fehler.

Lösung:

```
sudo apt-get install python-daemon python-watchdog
```

Überprüft das einfach mal.

Neuerstellung daemon-file

12. März 2016

Ausgangslage:

Bei der Installation wurde kein daemon-File erzeugt und / oder soll nachträglich erzeugt werden.

Achtung: Die Datei /etc/init.d/doorpi wird ohne Rückfrage überschrieben!

<https://www.doorpi.org/forum/thread/12-doorpi-als-daemon-dienst-service-autostart-unter-jessie-einrichten/?postID=33#post33>

```
cd /tmp
```

```
wget
```

```
https://raw.githubusercontent.com/motom001/DoorPi/master/doorpi/docs/service/create\_daemon\_file.py -O - | sudo python
```

```
sudo systemctl daemon-reload
```

```
sudo reboot
```

nach reboot kontrollieren

sudo systemctl daemon-reload

im putty Fenster dann das hier:

```
pi@Door:/tmp $ sudo systemctl daemon-reload
```

```
pi@Door:/tmp $ sudo systemctl daemon-reload
```

```
pi@Door:/tmp $ sudo /etc/init.d/doorpi start
```

```
[ ok ] Starting doorpi (via systemctl): doorpi.service.
```

```
pi@Door:/tmp $ sudo /etc/init.d/doorpi status
```

● doorpi.service - LSB: DoorPi

Loaded: loaded (/etc/init.d/doorpi)

Active: active (running) since Do 2017-05-04 22:50:19 CEST; 20s ago

Process: 4549 ExecStart=/etc/init.d/doorpi start (code=exited, status=0/SUCCESS)

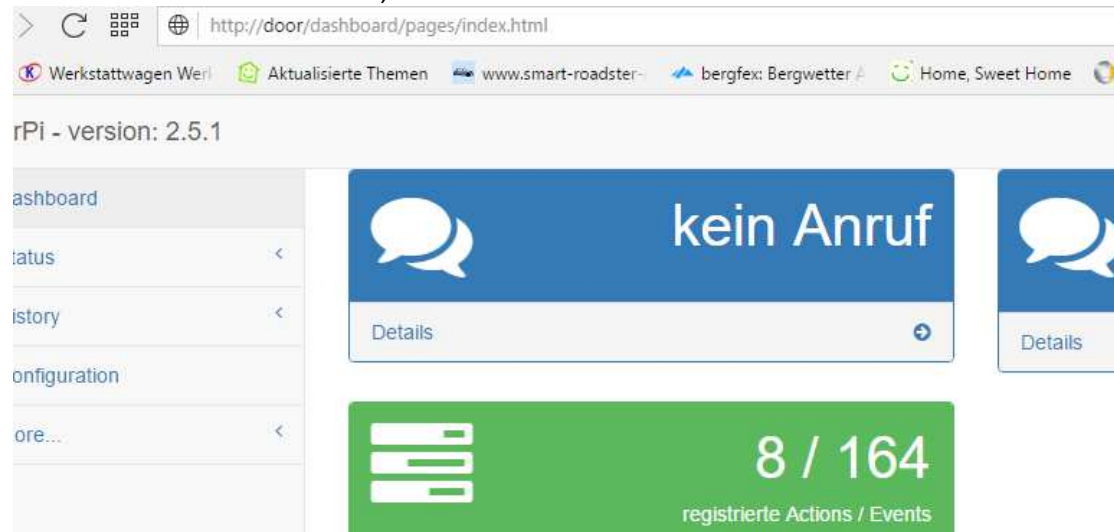
CGroup: /system.slice/doorpi.service

└─4557 /usr/bin/python /usr/local/bin/doorpi_cli start --configfile /usr/local/etc...

sudo reboot

dann im Win Fenster doorpi starten

danach Windows kontrollieren,



wenn es so aussieht ist alles ok

muss also noch in Autostart, Anleitung hier:

<https://www.doorpi.org/forum/thread/12-doorpi-als-daemon-dienst-service-autostart-unter-jessie-einrichten/?postID=18#post18>

Danach wird dem System dieser Daemon in den "Autostart" eingetragen:

Quellcode

```
sudo systemctl daemon-reload
```

```
sudo update-rc.d doorpi defaults
```

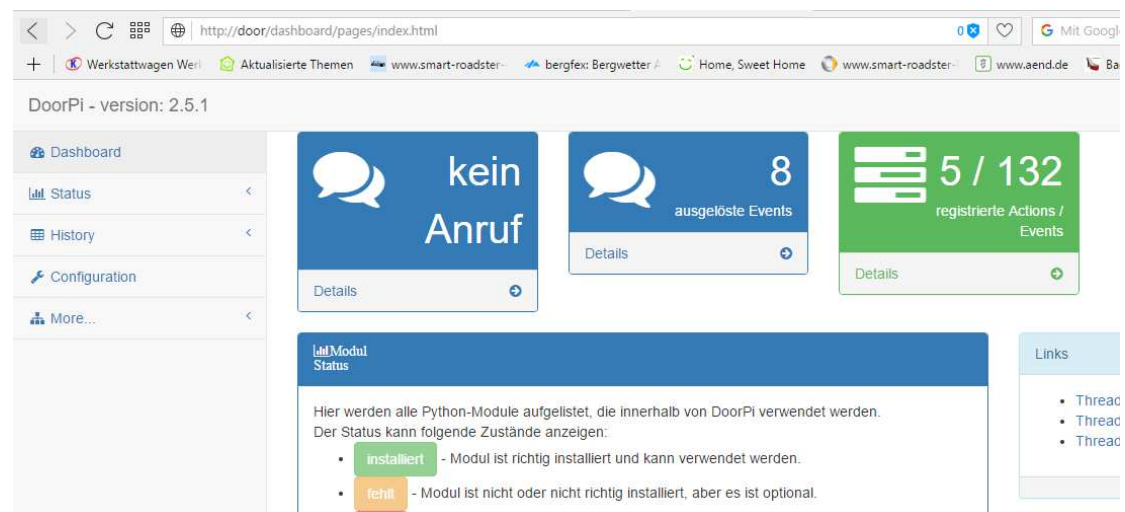
```
pi@Door:~ $ sudo systemctl daemon-reload
pi@Door:~ $ sudo update-rc.d doorpi defaults
pi@Door:~ $
```

Fertig - mit dem nächsten Systemstart wird DoorPi automatisch gestartet und bei einem System-Shutdown wird DoorPi ordentlich runtergefahren.

Test:

```
sudo reboot
```

Test im Windows direkt nach Neustart



Kamera einrichten

<https://raspberrypi.tips/faq/raspberry-pi-kamera-einrichten-videos-und-fotos-erstellen/>

`sudo apt-get -y update && sudo apt-get -y upgrade`

dann ein Testbild machen:

`pi@Door:~ $ raspistill -v -o test.jpg`

Raspi rattert, Ausgabe::

raspistill Camera App v1.3.11

Width 2592, Height 1944, quality 85, filename test.jpg

Time delay 5000, Raw no

Thumbnail enabled Yes, width 64, height 48, quality 35

Link to latest frame enabled no

Full resolution preview No

Capture method : Single capture

Preview Yes, Full screen Yes

Preview window 0,0,1024,768

Opacity 255

Sharpness 0, Contrast 0, Brightness 50

Saturation 0, ISO 0, Video Stabilisation No, Exposure compensation 0

Exposure Mode 'auto', AWB Mode 'auto', Image Effect 'none'

Metering Mode 'average', Colour Effect Enabled No with U = 128, V = 128

Rotation 0, hflip No, vflip No

ROI x 0.000000, y 0.000000, w 1.000000 h 1.000000

Camera component done

Encoder component done

Starting component connection stage

Connecting camera preview port to video render.

Connecting camera stills port to encoder input port

Opening output file test.jpg

Enabling encoder output port

Starting capture -1

Finished capture -1

Closing down

Close down completed, all components disconnected, disabled and destroyed

mit ls-1 kontrollieren ob Datei angelegt wurde:

pi@Door:~ \$ ls -l

setuptools-33.1.1.zip

test.jpg

also Bild vorhanden

Raspberry Pi Kamera über das Netzwerk Streamen, hier mit VLC, hab ich bei Jerssy lite so gemacht, bei Jessy pixel hier nicht, weiter mit mjpg streamer weiter unten !

Das Streaming des Kamera Signals ins Netzwerk erledigt der bekannte VLC Player für uns. Diesen installieren wir als erstes via apt

sudo apt-get install vlc

Der Trick ist jetzt das Video Signal von raspivid an den VLC-Player zu übergeben (das macht die Pipe | und der Parapeter -o – für uns) und von diesem einen Stream bereitzustellen. cvlc startet die Kommandozeilen-Version von VLC.

raspivid -o - -t 0 -n | cvlc -vvv stream:///dev/stdin --sout '#rtp{sdp=rtsp://:8554/}' :demux=h264

Natürlich könnt ihr raspivid auch noch weitere Parameter übergeben, zum Beispiel wenn ihr die Auflösung oder die maximalen Bilder pro Sekunde begrenzen wollt.

raspivid

“-o -” Schreibt die Video Ausgabe von raspivid in den std out

“-t 0” Deaktiviert das Timeout

“-n” Deaktiviert den Preview Modus am RasPi

cvlc

“-vvv” gibt an wohler VLC den Stream holen soll

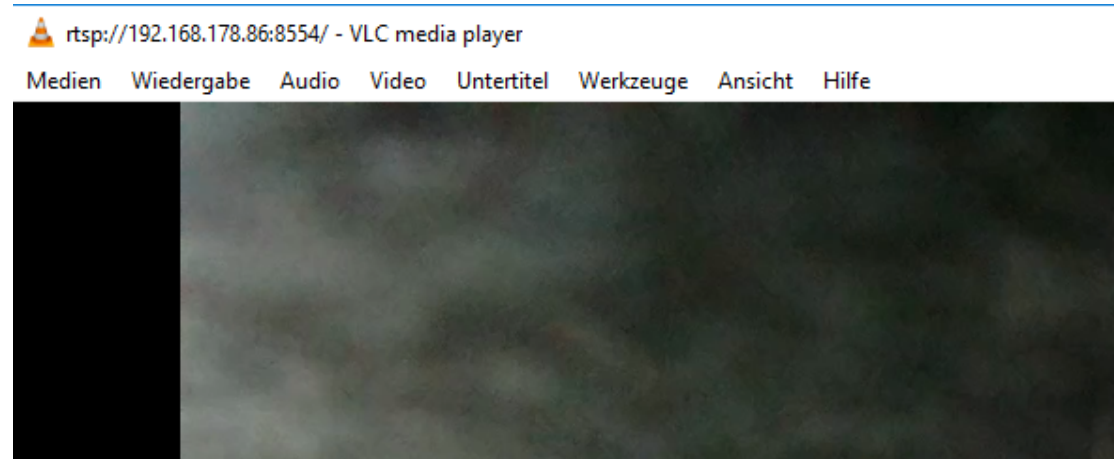
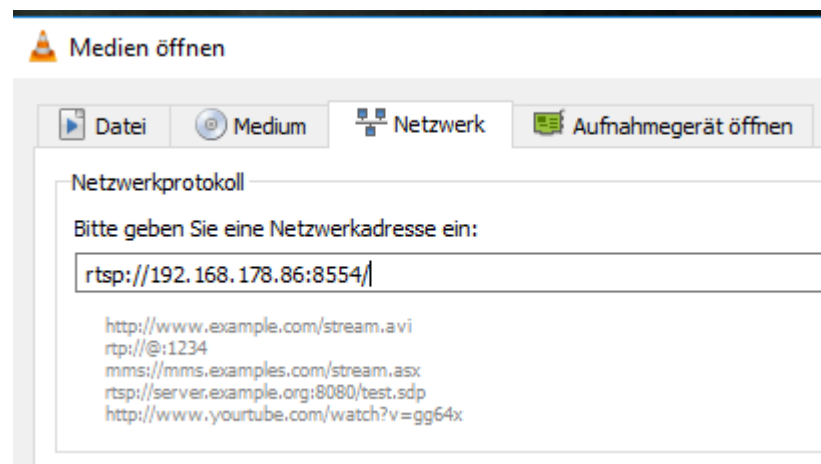
“-sout” gibt an wohin der Stream ausgegeben werden soll

Zum Anzeigen des Video Stream könnt ihr auf einem beliebigen Rechner (im selben Netzwerk) nun VLC ausführen und zum Beispiel via Windows über “Medien” > “Netzwerkstream öffnen” die Wiedergabe starten. Als Adresse gibt ihr die IP-Adresse eures Raspberry Pi im folgenden Format an.

rtsp://192.168.178.27:8554/

Natürlich müsst ihr 192.168.178.27 mit der IP eures RasPi ersetzen.

VLC Player öffnen,Adresse eingeben



im putty Fenster

```
[b3d03df0] core art finder debug: no art finder modules matched
[008a48f8] core libvlc debug: art not found for stdin
[00935960] core stream out debug: net: connecting to [192.168.178.55]:64072
[00935960] core stream out debug: net: connecting to [192.168.178.55]:64073 from
[192.168.178.86]:43292
```

also Bild wird inVLC angezeigt Kamera läuft

mjpg-streamer Installation

Installation des mjpg-streamer.

15. März 2016

<https://www.doorpi.org/forum/thread/36-installation-des-mjpg-streamer/?postID=6271&highlight=MJPEG-Streamer#post6271>

Quellcode

`sudo apt-get -y update && sudo apt-get -y upgrade`

`sudo apt-get install build-essential libjpeg-dev imagemagick subversion libv4l-dev
checkinstall`

```
netpbm (2:10.0-15.2) wird eingerichtet ...  
subversion (1.8.10-6+deb8u4) wird eingerichtet ...  
Trigger für libc-bin (2.19-18+deb8u7) werden verarbeitet ...  
pi@Door:~ $
```

`sudo modprobe bcm2835-v4l2`

Sollte es hier zu Problemen kommen ist wahrscheinlich die Kamera nicht aktiviert.

daher mit „sudo raspi-config“ ins Konfig und da nochmals aktivieren

weiter gehts mit:

`raspistill -o /tmp/test.jpg`

```
pi@Door:~ $ cd /tmp/  
pi@Door:/tmp $ ls -l  
pulse-DJkzTEkJgm4i  
pulse-PKdhtXMmrl8n  
ssh-UFtlFFaSBviE  
ssh-UKwU9DGbBA14  
systemd-private-702277bb10f04  
test.jpg  
pi@Door:/tmp $
```

`cd`

`svn checkout svn://svn.code.sf.net/p/mjpg-streamer/code/ mjpg-streamer-code`


```
A      mjpg-streamer-code/uvnc-streamer/Makefile
A      mjpg-streamer-code/.deps
U      mjpg-streamer-code
Ausgecheckt, Revision 182.
pi@Door:~ $
pi@Door:~ $
```

`cd mjpg-streamer-code/mjpg-streamer`

mit diesem Befehl landet man auf jeden Fall im richtigen Verzeichnis, war bei mir mal ein Problem beim ganzen probieren, daher hiermal eingefügt

`cd /home/pi/mjpg-streamer-code/mjpg-streamer`

Als nächstes erstellen wir einen Patch.

Quellcode

`sudo nano input_uvc_patch`

Den Inhalt wie im Post beschrieben

<https://www.doorpi.org/forum/thread/36-installation-des-mjpg-streamer/?postID=6271&highlight=MJPEG-Streamer#post6271>

einfügen und mit Srtg X abspeichern

dann sollte mit `ls -l` das hier rauskommen:

```
pi@Door:~/mjpg-streamer-code/mjpg-streamer $ cd /home/pi/mjpg-streamer-code/mjpg-streamer
pi@Door:~/mjpg-streamer-code/mjpg-streamer $ sudo nano input_uvc_patch
pi@Door:~/mjpg-streamer-code/mjpg-streamer $ ls -l
CHANGELOG
input_uvc_patch
LICENSE
Makefile
mjpg_streamer.c
mjpg_streamer.h
plugins
README
scripts
start.sh
TODO
utils.c
utils.h
www
pi@Door:~/mjpg-streamer-code/mjpg-streamer $
```

direkt danach gehts weiter mit dem hier:

<https://www.doorpi.org/forum/thread/723-mjpg-streamer-geht-nicht-im-mjpg-format/?postID=6442#post6442>

Sirlcy Anfänger Mittwoch 17:40 , hat mir eine kleine Anleitung geschrieben, was der patch von ihm genau macht ?? (bekomme ich auch noch raus)

Ich versuche mal eine kleine Anleitung:

1. Die Datei (input_uvc_patch.txt) im Anhang runterladen oder das blaue kopieren, in notepad++ in windows reinkopieren, dann mit dem Namen input_uvc_patch.txt abspeichern und dann mit filezilla zum raspi kopieren. (blauer Text)

```
--- plugins/input_uvc/input_uvc.c    (revision 174)
+++ plugins/input_uvc/input_uvc.c    (working copy)
@@ -405,9 +405,13 @@

    if(pcontext->videoin->formatIn == V4L2_PIX_FMT_YUYV) {

        DBG("compressing frame from input: %d\n", (int)pcontext->id);

        pglobal->in[pcontext->id].size = compress_yuyv_to_jpeg(pcontext->videoin,
pglobal->in[pcontext->id].buf, pcontext->videoin->framesizeIn, gquality);
+
+    /* copy this frame's timestamp to user space */
+
+    pglobal->in[pcontext->id].timestamp = pcontext->videoin->buf.timestamp;
    } else {

        DBG("copying frame from input: %d\n", (int)pcontext->id);

-    pglobal->in[pcontext->id].size = memcpy_picture(pglobal->in[pcontext->id].buf,
pcontext->videoin->tmpbuffer, pcontext->videoin->buf.bytesused);
+
+    pglobal->in[pcontext->id].size = memcpy_picture(pglobal->in[pcontext->id].buf,
pcontext->videoin->tmpbuffer, pcontext->videoin->tmpbytesused);
+
+    /* copy this frame's timestamp to user space */
+
+    pglobal->in[pcontext->id].timestamp = pcontext->videoin->tmptimestamp;
    }

    #if 0

@@ -418,8 +422,6 @@

    prev_size = global->size;

#endif

-    /* copy this frame's timestamp to user space */
-
-    pglobal->in[pcontext->id].timestamp = pcontext->videoin->buf.timestamp;

    /* signal fresh_frame */
```

```

        pthread_cond_broadcast(&pglobal->in[pcontext->id].db_update);
Index: plugins/input_uvc/v4l2uvc.c
=====
--- plugins/input_uvc/v4l2uvc.c (revision 174)
+++ plugins/input_uvc/v4l2uvc.c (working copy)
@@ -450,6 +450,8 @@
    */

    memcpy(vd->tmpbuffer, vd->mem[vd->buf.index], vd->buf.bytesused);
+   vd->tmpbytesused = vd->buf.bytesused;
+   vd->tmptimestamp = vd->buf.timestamp;

    if(debug)
        fprintf(stderr, "bytes in used %d \n", vd->buf.bytesused);
Index: plugins/input_uvc/v4l2uvc.h
=====
--- plugins/input_uvc/v4l2uvc.h (revision 174)
+++ plugins/input_uvc/v4l2uvc.h (working copy)
@@ -28,6 +28,7 @@

#include <stdio.h>
#include <stdint.h>
#include <string.h>
#include <fcntl.h>
#include <unistd.h>
@@ -105,6 +106,8 @@

int framecount;

```

```

int recordstart;

int recordtime;

+ uint32_t tmpbytesused;

+ struct timeval tmptimestamp;

};

```

Datei in Notepad kopieren und speichern mit Endung txt (s.o.)

2. Die Datei auf den Pi übertragen ins Verzeichnis "mjpg-streamer-code/mjpg-streamer".
Dazu kann man z.B. FileZilla nehmen.

```

pi@Door:~/mjpg-streamer-code/mjpg-streamer $ ls -l
CHANGELOG
input_uvc_patch
input_uvc_patch.txt
LICENSE

```

3. Die alte Patch-Datei mit der neuen überschreiben:

`sudo mv input_uvc_patch.txt input_uvc_patch`

```

pi@Door:~/mjpg-streamer-code/mjpg-streamer $ sudo mv input_uvc_patch.txt input_uvc_patch
pi@Door:~/mjpg-streamer-code/mjpg-streamer $ ls -l
CHANGELOG
input_uvc_patch
LICENSE

```

4. Dann den Patch ausführen mit:

`patch -p0 < input_uvc_patch`

```

pi@Door:~/mjpg-streamer-code/mjpg-streamer $ patch -p0 < input_uvc_patch
patching file plugins/input_uvc/input_uvc.c
patching file plugins/input_uvc/v4l2uvc.c
patching file plugins/input_uvc/v4l2uvc.h
Hunk #2 succeeded at 106 with fuzz 1.
pi@Door:~/mjpg-streamer-code/mjpg-streamer $

```

5. Weiter in der Anleitung "Installation Software MJPG-Streamer"

<https://www.doorpi.org/forum/thread/36-installation-des-mjpg-streamer/?postID=6271&highlight=MJPG-Streamer#post6271>

Wir führen anschließend diese Befehle aus und verlassen danach das Verzeichnis wieder.

Quellcode

make USE_LIBV4L2=true clean all

dauert wieder eine ganze Zeit

```
make[1]: Leaving directory '/home/pi/mjpg-streamer-code/mjpg-streamer/plugins/input_file'
cp plugins/input_testpicture/input_testpicture.so .
make -C plugins/input_file all
make[1]: Entering directory '/home/pi/mjpg-streamer-code/mjpg-streamer/plugins/input_file'
gcc -O2 -DLINUX -D_GNU_SOURCE -Wall -shared -fPIC -o input_file.so input_file.c
make[1]: Leaving directory '/home/pi/mjpg-streamer-code/mjpg-streamer/plugins/input_file'
cp plugins/input_file/input_file.so .
pi@Door:~/mjpg-streamer-code/mjpg-streamer $
```

sudo make DESTDIR=/usr/local install

```
make[1]: Leaving directory '/home/pi/mjpg-streamer-code/mjpg-streamer/plugins/input_file'
cp plugins/input_file/input_file.so .
pi@Door:~/mjpg-streamer-code/mjpg-streamer $ sudo make DESTDIR=/usr/local install
install --mode=755 mjpg_streamer /usr/local/bin
install --mode=644 input_uvc.so output_file.so output_udp.so output_http.so input_testpicture.so input_file.so /usr/local/lib/
install --mode=755 -d /usr/local/www
install --mode=644 -D www/* /usr/local/www
pi@Door:~/mjpg-streamer-code/mjpg-streamer $ cd
pi@Door:~ $
```

cd

Nun geben wir folgendes ein, nur um zu schauen das auch alles richtig installiert wurde.

Quellcode

sudo modprobe bcm2835-v4l2

Hier dürfen keine Fehlermeldungen erscheinen.

```
pi@Door:~/mjpg-streamer-code/mjpg-streamer $ cd
pi@Door:~ $ sudo modprobe bcm2835-v4l2
pi@Door:~ $
```

Jetzt kann der Streamer gestartet werden, mit folgendem Befehl:

Quellcode

```
mjpg_streamer -i "/usr/local/lib/input_uvc.so -d /dev/video0 -n -r 1024x768 -f 24 -q 80" -o
"/usr/local/lib/output_http.so -n -w /usr/local/www -p 9000"
```

Hier wird ein kleiner Webserver gestartet der unter Deiner Adresse des raspberypi zu erreichen ist aber mit der Port Nummer 9000.

Bsp.:

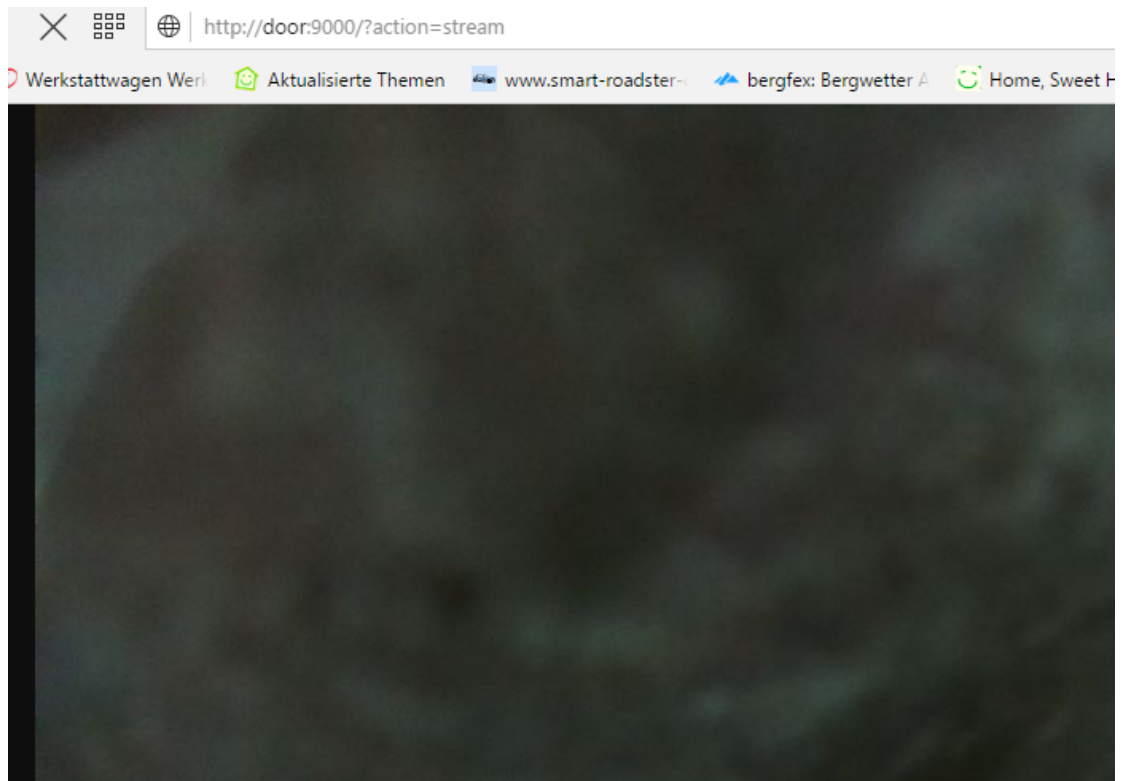
192.168.178.200:9000



```
pi@Door:~ $ mjpg_streamer -i "/usr/local/lib/input_uvc.so -d
ib/output_http.so -n -w /usr/local/www -p 9000"
MJPEG Streamer Version: svn rev: 3:172M
i: Using V4L2 device.: /dev/video0
i: Desired Resolution: 1024 x 768
i: Frames Per Second.: 24
i: Format.....: MJPEG
o: www-folder-path...: /usr/local/www/
o: HTTP TCP port.....: 9000
o: username:password.: disabled
o: commands.....: disabled
```

oder im win explorer mit dem angegeben link:

<http://door:9000/?action=stream>



mit

str c

beendet man den streamer im putty fenster

```
0: Commands.....: disabled
^Csetting signal to stop
i: cleaning up resources allocated by input thread
force cancellation of threads and cleanup resources
o: cleaning up resources allocated by server thread #00
done
pi@Door:~ $
```

Wenn man den mjpg_streamer beim booten automatisch gestartet haben möchte der sollte folgende Datei anlegen:

Quellcode

```
sudo nano /etc/init.d/mjpg_streamer
```

das was nun folgt in den Editor einfügen (alles was blau ist) in der Beschreibung aber zum kopieren bereitgestellt

```
#!/bin/sh
```

```
# /etc/init.d/mjpg_streamer
```

```
### BEGIN INIT INFO
```

```
# Provides:      mjpg_streamer
```

```
# Required-Start: $all
```

```
# Required-Stop:  $all
```

```
# Default-Start:  2 3 4 5
```

```
# Default-Stop:   0 1 6
```

```
# Short-Description: MJPG_Streamer_autostart
```

```
### END INIT INFO
```

```
start()
```

```
{
```

```
modprobe bcm2835-v4l2
```

```
sleep:2
```

```
echo "Starting mjpg-streamer..."
```

```
/usr/local/bin/mjpg_streamer -i "/usr/local/lib/input_uvc.so -d /dev/video0 -n -r  
1024x768 -f 24 -q 80" -o "/usr/local/lib/output_http.so -n -w /usr/local/www -p  
9000" >/dev/null 2>&1 &
```

```
}
```

```
stop()
```

```
{
```

```
echo "Stopping mjpg-streamer..."
```

```
kill -9 $(pidof mjpg_streamer) >/dev/null 2>&1
```

```
}
```

```
case "$1" in
```

```
start)
```



```

start

;;

stop)

stop

;;

restart)

stop

start

;;

*)

echo "Usage: $0 {start|stop|restart}"

;;

esac

exit 0

```

```

GNU nano 2.2.6      Datei: /etc/init.d/mjpg_streamer      Verändert
echo "Stopping mjpg-streamer..."
kill -9 $(pidof mjpg_streamer) >/dev/null 2>&1
}

case "$1" in
start)
start
;;
stop)
stop
;;
restart)
stop
start
;;
*)
echo "Usage: $0 {start|stop|restart}"
;;
esac

exit 0

```

[^]G Hilfe [^]C Speichern [^]R Datei öffnen [^]Y Seite zurück [^]X Ausschneiden [^]Q Cursor
[^]N Beenden [^]U Ausrichten [^]W Wo ist [^]V Seite vor [^]U Ausschn. rückgän [^]T Rechtschr.

mit str x speichern und dann mit "j" bestätigen

das hier nicht vergessen:

sudo chmod +x /etc/init.d/mjpg_streamer

Mit diesem Befehl wird die eben erstellte Autostart-Datei in den Autostart eingetragen:

Quellcode

`sudo update-rc.d mjpg_streamer defaults`

Danach kannst Du es bequem per:

`sudo service mjpg_streamer start`

starten.

Beenden kannst Du es mit:

`sudo service mjpg_streamer stop`

Kontrolle

`sudo service mjpg_streamer status`

```
root@debian:~# sudo service mjpg_streamer status
mjpg_streamer.service - LSB: MJPG_Streamer_autostart
Loaded: loaded (/etc/init.d/mjpg_streamer)
Active: active (running) since Mi 2017-05-03 18:51:03 CEST; 16min ago
Process: 11432 ExecStart=/etc/init.d/mjpg_streamer start (code=exited, status=0/SUCCESS)
CGroup: /system.slice/mjpg_streamer.service
└─11435 /usr/local/bin/mjpg_streamer -i /usr/local/lib/input_uvc.so -d /dev/video0 -n -r 1024x768

Mai 03 18:51:02 Door mjpg_streamer[11435]: MJPG-streamer [11435]: Desired Resolution: 1024 x 768
Mai 03 18:51:02 Door mjpg_streamer[11435]: MJPG-streamer [11435]: Frames Per Second.: 24
Mai 03 18:51:02 Door mjpg_streamer[11435]: MJPG-streamer [11435]: Format.....: MJPEG
Mai 03 18:51:02 Door mjpg_streamer[11435]: MJPG-streamer [11435]: www-folder-path...: /usr/local/www/
Mai 03 18:51:02 Door mjpg_streamer[11435]: MJPG-streamer [11435]: HTTP TCP port.....: 9000
Mai 03 18:51:02 Door mjpg_streamer[11435]: MJPG-streamer [11435]: username:password.: disabled
Mai 03 18:51:02 Door mjpg_streamer[11435]: MJPG-streamer [11435]: commands.....: disabled
Mai 03 18:51:02 Door mjpg_streamer[11435]: MJPG-streamer [11435]: starting input plugin /usr/local/lib/i
Mai 03 18:51:02 Door mjpg_streamer[11435]: MJPG-streamer [11435]: starting output plugin: /usr/local/lib
Mai 03 18:51:03 Door systemd[1]: Started LSB: MJPG_Streamer_autostart.
Hint: Some lines were ellipsized, use -l to show in full.
```

Reboot:

Kontrolle ob auch alles richtig startet

`sudo service mjpg_streamer status`

alles idetisch wie oben :-)

bis hierher neu !