

# Smart MediCine Installationsanleitung

## Inhalt

1.	Verwendete Hard- und Software .....	2
2.	Grundlegende Konfigurationen.....	2
2.1.	Installation von Raspian .....	2
2.2.	Konfiguration von Raspbian .....	3
2.3.	(Optional) Wlan Dongle installieren .....	3
2.4.	Voreinstellungen zur Verwendung von Winscp .....	3
2.5.	OpenHAB installieren .....	4
3.	Einrichten von Smart Medicine .....	4
3.1.	Webapp einrichten .....	4
3.2.	Grundlegende Konfigurationen.....	5
3.3.	Sensoren einrichten .....	9

## 1. Verwendete Hard- und Software

### Software

Software	Zugangsdaten/Zusätzliche Details
Google Kalender	<b>USER:</b> <a href="mailto:smartmedicine06@gmail.com">smartmedicine06@gmail.com</a> <b>PW:</b> smarthome06

### Hardware

Hardware	Bezugsquelle
Raspberry Pi 2	<a href="https://www.amazon.de/Raspberry-Pi-quad-core-Cortex-A7-compatibility/dp/B00T2U7R7I/ref=sr_1_3?ie=UTF8&amp;qid=1469227102&amp;sr=8-3&amp;keywords=raspberry+pi+2">https://www.amazon.de/Raspberry-Pi-quad-core-Cortex-A7-compatibility/dp/B00T2U7R7I/ref=sr_1_3?ie=UTF8&amp;qid=1469227102&amp;sr=8-3&amp;keywords=raspberry+pi+2</a>
Hall Sensor	<a href="http://www.ebay.de/itm/like/251960684676?lpid=106&amp;chn=ps&amp;ul_noapp=true">http://www.ebay.de/itm/like/251960684676?lpid=106&amp;chn=ps&amp;ul_noapp=true</a>
Tilt Sensor	<a href="http://www.ebay.de/itm/Tilt-Sensor-Module-Vibration-Sensor-for-Arduino-STM32-AVR-Raspberry-Pi-DE-/331304858618?hash=item4d234ef3fa:g:yqgAAOSw-jhUAKHr">http://www.ebay.de/itm/Tilt-Sensor-Module-Vibration-Sensor-for-Arduino-STM32-AVR-Raspberry-Pi-DE-/331304858618?hash=item4d234ef3fa:g:yqgAAOSw-jhUAKHr</a>
Jumper Kabel (Female – Female)	<a href="http://www.ebay.de/itm/40x-30cm-Female-Female-jumper-wire-Cable-Kabel-Steckbrücken-Drahtbrücken-Arduino-/161114579903?hash=item25832d63bf:g:K6oAAOSwU9xUUHew">http://www.ebay.de/itm/40x-30cm-Female-Female-jumper-wire-Cable-Kabel-Steckbrücken-Drahtbrücken-Arduino-/161114579903?hash=item25832d63bf:g:K6oAAOSwU9xUUHew</a>
Magnete	<a href="https://www.amazon.de/gp/product/B000VKFT6O/ref=oh_aui_detailpage_o01_s00?ie=UTF8&amp;psc=1">https://www.amazon.de/gp/product/B000VKFT6O/ref=oh_aui_detailpage_o01_s00?ie=UTF8&amp;psc=1</a>
Pillendose	Handelsübliche Pillendosen

## 2. Grundlegende Konfigurationen

### 2.1. Installation von Raspian

1. Raspbian von <https://www.raspberrypi.org/downloads/> runterladen und entpacken
2. Dann Winscp von [http://www.chip.de/downloads/Win32-Disk-Imager\\_46121030.html](http://www.chip.de/downloads/Win32-Disk-Imager_46121030.html) runterladen und öffnen.
3. Putty von <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html> runterladen und installieren.
4. Micro SD-Karte in den Computer stecken.
5. Bei Image-File die Raspbian ISO-Datei auswählen, das Verzeichnis der Micro SD-Karte wählen und auf „Write“ klicken.
6. Wenn das Betriebssystem erfolgreich installiert wurde, dann muss die SD-Karte in den Raspberry gesteckt werden.
7. Darauf achten, dass der Raspberry mit dem Internet verbunden ist.
8. Putty starten und mit der IP des Raspberry verbinden. User ist „PI“ und Passwort ist „Raspberry“ (Die IP des Raspberry kann z. B. mit **Advanced IP Scanner** herausgefunden werden)

## 2.2. Konfiguration von Raspbian

Nachdem Raspbian erfolgreich installiert wurde, müssen noch einige Einstellungen vorgenommen werden. Unter anderem muss der Speicher, der standardmäßig nicht den kompletten SD-Karten speicher verwendet, erweitert werden.

Diesbezüglich geht man folgendermaßen vor:

1. Mit putty sich mit dem Raspberry verbinden
2. `sudo raspi-config` aufrufen und „Expand File-System“ auswählen.
3. Daraufhin bestätigen.

Nun sollte das System neustarten und der Speicherplatz erweitert worden sein.

Desweiteren muss das System auf den aktuellen Stand gebracht werden:

```
sudo apt-get update
```

Nachdem das update durchgeführt wurde muss ein upgrade durchgeführt werden

```
sudo apt-get upgrade
```

Wenn das Update und Upgrade erfolgreich durchgeführt wurde, dann muss noch Java installiert werden:

```
sudo apt-get install oracle-java7-jdk
```

Ob Java erfolgreich installiert wurde kann folgendermaßen überprüft werden:

```
java -version
```

## 2.3. (Optional) Wlan Dongle installieren

```
sudo nano /etc/network/interfaces
```

```
auto lo
iface lo inet loopback
iface eth0 inet dhcp
auto wlan0
allow-hotplug wlan0
iface wlan0 inet dhcp
wpa-ap-scan 1
wpa-scan-ssid 1
wpa-ssid "DEIN-WLAN-NAME"
wpa-psk "DEIN-WLAN-SCHLÜSSEL"
```

```
sudo service networking restart
```

## 2.4. Voreinstellungen zur Verwendung von Winscp

Damit man mit Winscp Datenaustauschen kann, muss man dem Root-User ein Passwort vergeben:

```
Sudo passwd root
```

Danach das Passwort eingeben und bestätigen.

Nun muss nur noch RootLogin über SSH erlaubt werden. Dies kann in der folgendermaßen erfolgen:

```
sudo nano /etc/ssh/sshd_config
```

Die folgende Zeile auskommentieren:

```
PermitRootLogin without-password
```

Und den folgenden Befehl hinzufügen:

```
PermitRootLogin yes
```

Danach SSH und Raspberry neustarten

```
sudo service ssh restart  
sudo reboot
```

Nun kann man sich über mit Winscp auf den Raspberry verbinden, wenn man eine aktive SSH-Verbindung besteht. (zum Beispiel durch Putty)

## 2.5. OpenHAB installieren

Die OpenHAB-Dateien können mit WINSCP über SSH auf den Raspbery geladen werden. Dafür geht man folgendermaßen vor:

1. Winscp starten
2. In das Verzeichnis /opt navigieren
3. Den Openhab Ordner hochladen
4. In putty in das Verzeichnis /opt/openhab navigieren
5. Den Befehl `sudo chmod +x start.sh` ausführen, damit openhab auch gestartet werden kann.
6. Dann kann man OpenHAB mit dem Befehl `sudo ./start.sh` starten
7. Im Browser die URL `http://ipVonRaspberry:8080/openhab.app?sitemap=demo` eingeben
8. Nun sollte die OpenHAB UI sichtbar sein

## 3. Einrichten von Smart Medicine

### 3.1. Webapp einrichten

1. Den Ordner smartmedicine aus dem App-Ordner in den webapps Ordner von dem zentralen OpenHAB kopieren.
2. OpenHAB starten und <http://<ip>:8080/smartmedicine/#> öffnen
3. Beim dem ersten Aufruf der App muss man sich bei dem Google Kalender anmelden. Dies kann erfolgen, indem man auf den Button „Authorize“ klicken und sich mit den folgenden Daten anmelden: **USER:** [smartmedicine06@gmail.com](mailto:smartmedicine06@gmail.com) **PW:** smarthome06

## 3.2.Grundlegende Konfigurationen





### OpenHAB noch nicht auf dem Raspberry eingerichtet:

1. Den Ordner openhab1.8.3, welches sich in dem Ordner OpenHAB befindet, in das Raspberry Verzeichnis /opt kopieren.
2. OpenHAB starten und mit Kapitel 3.3 Sensoren einrichten fortfahren.

### OpenHAB bereits auf dem Raspberry eingerichtet:

Wenn eine OpenHAB Version bereits vorhanden ist, dann muss folgendermaßen vorgegangen werden:

1. Die folgenden Addons müssen in den Addons Ordner geladen werden:

 org.openhab.binding.caldav-command-1.8.3  
 org.openhab.binding.caldav-personal-1.8.3  
 org.openhab.binding.milight-1.8.3  
 org.openhab.io.caldav-1.8.3

### Die folgenden Einstellungen in der openhab.cfg vornehmen:

1. Einbinden von MiLight

```
##### Milight Binding
#####
#
# Host of the first Milight bridge to control
milight:bridge1.host=192.168.0.255
# Port of the bridge to control (optional, defaults to 50000)
milight:bridge1.port=8899
#
# Host of the second Milight bridge to control
#milight:<MilightId2>.host=
# Port of the bridge to control (optional, defaults to 50000)
#milight:<MilightId2>.port=
```

## 2. Caldav

```
##### CalDAV IO Binding
#####

#
# Used to connect to Cal DAV. All parameters are required.
# Path to the calendar
# caldavio:<calendarId>:url=
caldavio:smarthome:url=https://www.google.com/calendar/dav/smartmedicine06@gmail.com/events

# Username for the calendar
# caldavio:<calendarId>:username=
caldavio:smarthome:username=smartmedicine06@gmail.com

# caldavio:smarthome:username=admin

# Password for the calendar
# caldavio:<calendarId>:password=
caldavio:smarthome:password=smarthome06
#
# Reload interval unit is minutes.
# Defines how often the calendar should be reloaded from server.
# Default is 60 minutes
# caldavio:<calendarId>:reloadInterval=
caldavio:smarthome:reloadInterval=1

# This defines which events are relevant for execution. Unit is in minutes.
# Default is 1 Day (1440 minutes)
# caldavio:<calendarId>:preloadTime=
caldavio:smarthome:preloadTime=1440

# A caldav Server is just a webdav Server which list files. Some servers does not use the valid
timestamp for modifications.
# If your calendar does not provide correct timestamps you have to set this false.
# Default is true
# caldavio:<calendarId>:lastModifiedFileTimeStampValid=
#caldavio:smarthome:lastModifiedFileTimeStampValid=false
```

```

# SSL verification can be disabled, if you don't want to import the server certificate
# into the java keystore. This is just needed for self-signed certificates, where the
# certificate path cannot be verified. Default is false. Do not set to true if no SSL is used.
caldavio:smarthome:disableCertificateVerification=true


# Timezone for events which does not have a timeZone information.
# Normally this is not required
# caldavio:timeZone=
#caldavio:timeZone=Europe/Berlin


##### CalDAV Command Binding
#####

# see CalDAV IO Binding
# Used to execute commands if events starts or ends with an easy notation in the event description.
# commaseperated (e. g. openhab, anothercalendar)
# caldavCommand:readCalendars=<ids from caldav-io>
caldavCommand:readCalendars=smarthome


##### CalDAV Personal Binding
#####

# see CalDAV IO Binding
# Used to toggle switch items for presence. Switched to ON if an event in the calendar occurs.
# And back to OFF if the event ends.
# Can also be used to show upcoming or active events


# Which calendars should be used to detect presence (comma separated)
# caldavPersonal:usedCalendars=<ids from caldav-io>
caldavPersonal:usedCalendars=smarthome


# If the location of the event is one of this identifiers, the presence will not be changed.
# Can be used for events which are at home or are just reminders. (comma separated, optional)
# caldavPersonal:homeIdentifiers=

```

3. Die folgenden Items anlegen:

```
String Light_scene    "Scenes"
Color Light_scene_ColorSelect "Scene Selector" <colorwheel> (MiLight)  {milight="bridge1;6;rgb"}
Switch MilightSwitch    {milight="bridge1;6;whiteMode"}
```

4. Die folgenden rules anlegen:

```
import org.openhab.core.library.types.*
import org.openhab.core.persistence.*
import org.openhab.model.script.actions.*

rule "Light Scenes"
when
Item Light_scene received command
then
if (receivedCommand=="rot") {
    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(0),new
PercentType(100),new PercentType(100)))
    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(0),new
PercentType(100),new PercentType(100)))
    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(0),new
PercentType(100),new PercentType(100)))
    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(0),new
PercentType(100),new PercentType(100)))
    executeCommandLine("python @ @/opt/openhab1.8.3/configurations/scripts/red.py")
}
if (receivedCommand=="gruen") {
    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(120),new
PercentType(100),new PercentType(100)))
    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(120),new
PercentType(100),new PercentType(100)))
    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(120),new
PercentType(100),new PercentType(100)))
    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(120),new
PercentType(100),new PercentType(100)))
    executeCommandLine("python @ @/opt/openhab1.8.3/configurations/scripts/green.py")
}
if (receivedCommand=="blau") {
```



```

    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(240),new
PercentType(100),new PercentType(100)))

    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(240),new
PercentType(100),new PercentType(100)))

    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(240),new
PercentType(100),new PercentType(100)))

    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(240),new
PercentType(100),new PercentType(100)))

executeCommandLine("python@ @/opt/openhab1.8.3/configurations/scripts/blue.py")
}

if (receivedCommand=="magenta") {

    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(300),new
PercentType(100),new PercentType(100)))

    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(300),new
PercentType(100),new PercentType(100)))

    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(300),new
PercentType(100),new PercentType(100)))

    sendCommand(Light_scene_ColorSelect, new HSBType(new DecimalType(300),new
PercentType(100),new PercentType(100)))

executeCommandLine("python@ @/opt/openhab1.8.3/configurations/scripts/magenta.py")
}

end

```

5. Die Skripte red.py, blue.py, green.py und magenta.py in den scripts Ordner von OpenHAB laden

### 3.3. Sensoren einrichten

#### Sensoren an der Pillendose anbringen

1. Der Tilt Sensor kann außerhalb der Pillendose oder auch direkt in der Pillendose angebracht werden.
2. Magneten an der Deckelinnenseite anbringen.
3. Hall Sensor muss in der Pillendose angebracht werden, damit das magnetische Feld des Magneten, welcher an der Deckelinnenseite angebracht ist, gemessen werden kann. Dadurch kann festgestellt werden, ob die Pillendose offen oder geschlossen ist.

#### Anschließen der Sensoren von dem Beispielsszenario (Pillenfarbe Rot)

Die Abbildung 1 zeigt den Aufbau zur Steuerung einer roten Pillendose. Diese kann beispielhaft für die anderen Sensoren verwendet werden. Wenn andere GPIO-Pins Ausgänge verwendet werden

sollen, dann muss die Datei red.py geöffnet werden und an der Stelle HALL\_GPIO der Ausgang für den Hall Sensor definiert werden und bei TILT\_GPIO der Ausgang für den Neigungssensor.

HALL\_GPIO = 8

TILT\_GPIO = 10

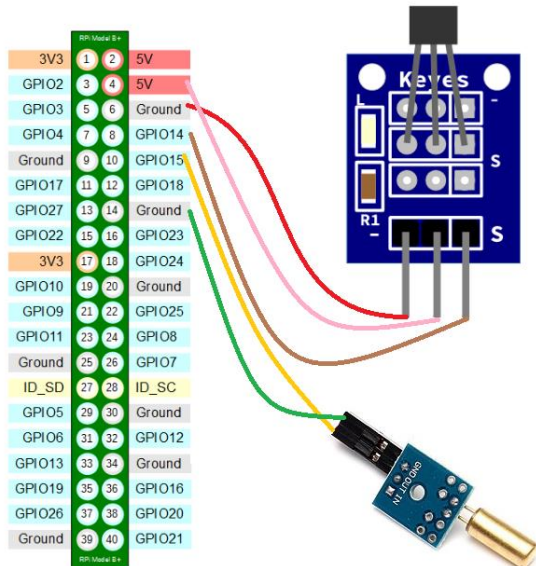


Abbildung 1: Beispielsanschluss für eine rote Pillendose

### Einrichten der weiteren Farben (blue, magenta bzw. green)

Wenn weitere Sensoren eingerichtet werden sollen, dann muss einfach das jeweilige Script dafür geöffnet werden und bei HALL\_GPIO der gewünschte Ausgang für den Hall Sensor definiert werden sowie bei TILT\_GPIO der gewünschte Ausgang für den TILT Sensor. Die restlichen Ground und 5V Anschlüsse können einfach an den freien Stellen angeschlossen werden.