Domoticz

What I did

LJM Dullaart

1. Intro

These are documentation sniplets of my Domoticz installation. It is not a coherent story on how to use Domiticz to automate your home. It is more a collection of ad-hoc things that I did.

2. Basic Set-up

My basic set-up is a Raspberry Pi with a number of additions. The main addition s are an RFXCOM RFXtrx433 that is used to send and receive the standard 433MHz IoT signals and a deCONZ USB Zigbee controller.

Domoticz is used as a sort of hub where everything comes together. It allows me to launch simple blockly scripts to react to events. Domoticz allows external scripts to update devices and launch scripts to do actual work. The web interface may not be intuitive, but it is easy to write a simple CGI script.

3. Smart meter

Due to Dutch regulations, I was forced to have a smart meter installed. The smart meter has a P1 port that allows the user to read the values. I bought a Homewizard Wi-Fi P1 Meter, which is a quite affordable one.

3.1 Activating the P1 meter

To be able to use the P1 meter, you need to install the Homewizard Energy app from the playstore or appstore. The app will be used the connect the device to the WiFi network.

The device uses DHCP to get an IP address. We need to access the device with `curl` so it is fairly important to assign a specific IP address. In my `dhcpd.conf` I have added:

```
host p1 {
hardware ethernet 3c:39:e7:25:ff:ff;
fixed-address 192.168.xxx.xxx;
}
```

On my DNS server, I have added the host `p1` with that address.

3.2 Connecting to Domoticz

It took quite a bit of puzzling before I could get the values of the p1 meter into Domoticz. I created a script that is launched from the crontab and updates 3 virtual sensors.

First, I added a new hardware.

For the hardware, I made three virtual sensors:

- slimme_meter_stroom type "Slimme meter elektra"
- slimme_meter_gas type "Gas"
- slimme_meter_active_power

The script below is used to get the data from the P1 meter and feed it, via the rest API to the virtual sensors.

```
#!/bin/bash
#REMOTE@ domoticz.home /usr/local/bin/p1_report
#########################################################################
# Set or change the hostnames below to match your environment
#########################################################################

DOMOTICZ='domoticz.home:8888'
P1=p1.home

#########################################################################
helpme(){
cat <<EOF

NAME:
          p1_report - report smart meter p1 values to Domoticz

SYNOPSIS:
          p1_report

DESCRIPTION:
P1_report reads the Homewizard Wi-Fi P1 meter and feeds the data into two
virtual sensors in Domoticz:
- slimme_meter_gas for the gas readings
- slimme_meter_stroom for the electicity reading
- slimme_meter_active_power
```

```
These three virtual sensors must be created manually in Domoticz.

The default URL for Domoticz is 'domoticz.home:8888' and the default
URL for the Homewizzard P1 meter is 'p1.home'. These can be changed in
the script to match your environment.

P1_report is typicaly started by cron to ensure a regular data feed to
Domoticz.

EOF
}

if [ "$1" = "-h" ] ; then
          helpme
          exit 0
fi

report=/tmp/p1_report.$$
tmp=/tmp/p1_reportmp.$$

date > /tmp/last_p1
now=$(date)

curl --silent  "http://$DOMOTICZ/json.htm?type=command&param=devices_list" |
sed 's/"//g;s/,//' |
while read nv col val ; do
          if [ "$nv" = 'name' ] ; then
                    name="$val"
          elif [ "$nv" = 'value' ] ; then
                    echo "$val $name"
          fi

done > $tmp

if curl -s $P1/api/v1/data > $report ; then
          :
else
          exit 0
fi

for line in $(cat $report| jq . |sed 's/^ *"//;s/,$//;s/": /:/p') ; do
          var=${line%:*}
          val=${line#*:}
          case $var in
                    (total_power_import_t1_kwh)   usage1=$(printf "%8.3f" $val | sed 's/[\. ]//g') ;;
                    (total_power_import_t2_kwh)   usage2=$(printf "%8.3f" $val | sed 's/[\. ]//g') ;;
                    (total_power_export_t1_kwh)   return1=$(printf "%8.3f" $val | sed 's/[\. ]//g') ;;
                    (total_power_export_t2_kwh)   return2=$(printf "%8.3f" $val | sed 's/[\. ]//g') ;;
                    (active_power_w)              active_power_w=$val ;;
                    (active_power_l1_w)           active_power_l1_w=$val ;;
                    (active_power_l2_w)           active_power_l2_w=$val ;;
                    (active_power_l3_w)           active_power_l3_w=$val ;;
                    (total_gas_m3)                          total_gas_m3=$(printf "%8.3f" $val | sed 's/[\. ]//g') ;;
                    (gas_timestamp)                         gas_timestamp=$val ;;
          esac
done

echo "$now usage1=$usage1 usage2=$usage2 return1=$return1 return2=$return2 active_power_w=$active_power_w total_gas_m3=$total_gas_m3" >> /tmp/p1.l

if [ $active_power_w -lt 0 ] ; then
          prod=$active_power_w
          cons=0
else
          prod=0
          cons=$active_power_w
fi

if [ $usage1 = 0 ] ; then
          if [ $usage2 = 0 ] ; then
                    if [ $return1 = 0 ] ; then
                              if [ $return2 = 0 ] ; then
                                        exit 0
                              fi
                    fi
          fi
fi


meter_idx=$(sed -n 's/ slimme_meter_stroom//p' $tmp)
if [ "$meter_idx" = "" ] ; then
          echo "No slimme meter">> /tmp/last_p1
```

```
else
          curl --silent  "http://$DOMOTICZ/json.htm?type=command&param=udevice&idx=$meter_idx&nvalue=0&svalue=$usage1;$usage2;$return1;$return2;$c
          echo "$usage1;$usage2;$return1;$return2;$cons;$prod" >> /tmp/last_p1
fi

meter_idx=$(sed -n 's/ slimme_meter_active_power//p' $tmp)
if [ "$meter_idx" = "" ] ; then
          echo "No slimme meter">> /tmp/last_p1
else
          curl --silent  "http://$DOMOTICZ/json.htm?type=command&param=udevice&idx=$meter_idx&nvalue=0&svalue=$active_power_w" > /dev/null
          echo "$active_power_w" >> /tmp/last_p1
fi

meter_idx=$(sed -n 's/ active_power//p' $tmp)
if [ "$meter_idx" = "" ] ; then
          echo "No slimme meter">> /tmp/last_p1
else
          curl --silent  "http://$DOMOTICZ/json.htm?type=command&param=udevice&idx=$meter_idx&nvalue=0&svalue=$active_power_w" > /dev/null
          echo "$active_power_w" >> /tmp/last_p1
fi

meter_idx=$(sed -n 's/ slimme_meter_gas//p' $tmp)
if [ "$meter_idx" = "" ] ; then
          echo "No gas meter">> /tmp/last_p1
else
          curl --silent  "http://$DOMOTICZ/json.htm?type=command&param=udevice&idx=$meter_idx&nvalue=0&svalue=$total_gas_m3" > /dev/null
          echo "$total_gas_m3" >> /tmp/last_p1
fi

tail -1024 /tmp/last_p1 > $tmp
mv $tmp /tmp/last_p1

rm -f $report $tmp
```

The script runs every minute from `cron` with the following crontab:

```
#USER root@domoticz.home                                                                                    #--p1.cron:root
#INSTALLEDFROM verlaine:src/domoticz                                                                        #--p1.cron:root
#  m  h  D   M  dow  cmd                                                                                    #--p1.cron:root
   *  *  *   *   *   /bin/bash /usr/local/bin/p1_report > /tmp/last_p1_report.out 2> /tmp/last_p1_report.err  #--p1.cron:root
# end p1                                                                                                    #--p1.cron:root
```

4. Somfy sunscreen

Somfy has a bridge that allows access access from Domoticz to the Somfy controlled sunscreens. That is a possible solution. But I had some trouble buying one, although availability now seems much better.

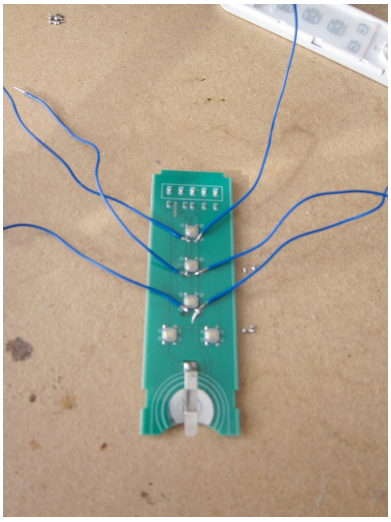
There is an alternative solution for those who are not afraid to use the soldering iron and do some programming.

4.1 Hardware

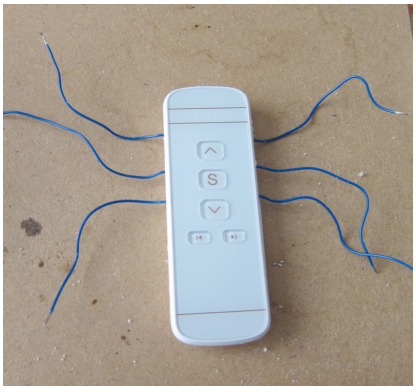
The idea is to control the buttons of a remote with relay. For this, a modified remote is needed and a USB relay card.

There are a number of relatively cheap Somfy remote clones available. I chose one from `"https://123afstandsbediening.nl/"`

I soldered wires on the contact points of the switches.

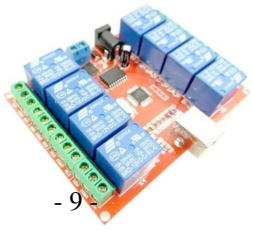


I made some holes in the sides of the remote to pull-out the wires.



A small blob of thermocol on the wires makes sure that the printed circuit board does not com under stress when handling the wires, I can now control the sunscreen up, down and stop by touching the appropriate wires.

The wires are connected to a USB relay card. I got a simple, 8 relay card from SOS solutions. I'll be using only 3 relay, but the price difference with the 4 relay card is negligible and it allows me to use the relay for other projects.



Connect the wires to the relays so that, when a relay is closed, a button is pushed, and when a relay opens, the button is released. I connected the down button to relay 1, stop to relay 2 and up to relay 3.

4.2 Software

4.2.1 usbrelay

First, the relay must be controllable from the Pi. `usbrelay` is an installable package, and if the relay card is in the right state, this is probably the easiest way.

However, if the card is in a wrong state, for example no name is given, then the latest version of `usbrelay` is required. That means compiling from source.

First, install the dependencies:

```
sudo apt install libhidapi-hidraw0 libhidapi-libusb0
sudo apt-get install python-dev cython libudev-dev libusb-1.0-0 libusb-dev libusb-1.0-0-dev libhidapi-dev libavahi-compat-libdnssd-dev
```

Then, get the latest version:

```
git clone https://github.com/darrylb123/usbrelay
```

And then compile:

```
cd usbrelay/
make
sudo make install
```

Because my card had no name attached to it, I had to give it its name.

```
usbrelay -d
```

gives information about the connected board, in my case:

```
libusbrelay: 1.2.1-4-gf858f1ad9f
usbrelay: 1.2.1-4-gf858f1ad9f
enumerate_relay_boards()Found 1 devices
Device Found
  type: 16c0 05df
  path: /dev/hidraw0
  serial_number:
Manufacturer: www.dcttech.com
  Product:      USBRelay8
  Release:      100
  Interface:    0
  Number of Relays = 8
  Module_type = 1
_1=0
_2=0
_3=0
_4=0
_5=0
_6=0
_7=0
_8=0
```

The serial number is missing, so that needs to be set. That can be done with

```
sudo ./usbrelay /dev/hidraw0=ZAA
```

or

```
sudo ./usbrelay /dev/hidraw0_0=ZAA
```

depending on what works for your card.

After that, `usbrelay -d` gives

```
libusbrelay: 1.2.1-4-gf858f1ad9f
usbrelay: 1.2.1-4-gf858f1ad9f
enumerate_relay_boards()Found 1 devices
Device Found
  type: 16c0 05df
  path: /dev/hidraw0
  serial_number: ZAA
Manufacturer: www.dcttech.com
  Product:      USBRelay8
  Release:      100
  Interface:    0
  Number of Relays = 8
  Module_type = 1
ZAA_1=0
ZAA_2=0
ZAA_3=0
ZAA_4=0
ZAA_5=0
ZAA_6=0
ZAA_7=0
ZAA_8=0
```

4.2.2 Domoticz

Thanks to a plugin, it is possible to launch a script with a slider for the sunscreens.

A Dummy-hardware is used to group the Somfy devices. I called it Somfy-virt. When the hardware is created, virtual sensors can be created. I created one with the type "schakelaar" (switch). It then appears under the tab "Schakelaars" and can be configured from there.

The On and Off actions refer to a script that is under the `domoticz/scripts` directory. The argument, here 2209, is the value of the Idx. That is required because the script needs to retrieve a value from domoticz.

4.2.3 The script

The script I use is as follows.

```
#!/bin/bash


index="$1"
DOMOTICZ='domoticz.home:8888'
VAR=/home/pi/domoticz/var
VALUE=$VAR/value.$index
tdown=20
```

```
now=$(date +%s)

sw(){
          rel=$1
          usbrelay ZAA_$rel=1
          sleep 0.1
          usbrelay ZAA_$rel=0
}

mkdir -p $VAR
if [ ! -f $VALUE ] ; then
          echo "$now 0" >  $VALUE
fi

read prevtime prevvalue < $VALUE

timeago=$((now-prevtime))

nieuwpct=$(curl -s "http://$DOMOTICZ/json.htm?type=devices&rid=${index}" | jq -r '.result[0].Level')

if [ $prevvalue -ge $nieuwpct ] ; then
          direction=up
          diff=$((prevvalue-nieuwpct))
else
          direction=down
          diff=$((nieuwpct-prevvalue))
fi

if [ "$nieuwpct" -le 5 ] ; then
          nieuwpct=0
elif [ "$nieuwpct" -ge 95 ] ; then
          nieuwpct=100
fi

if [ "$nieuwpct" = 0 ] ; then
          sw 3
          nieuwpct=0
elif [ "$nieuwpct" = 100 ] ; then
          sw 1
          nieuwpct=100
else
          t=$(($diff/4))
          if [ "$direction" = "up" ] ; then
                    sw 3
                    sleep $t
                    sw 2
          else
                    sw 1
                    sleep $t
                    sw 2
          fi

fi

echo "$now $nieuwpct"  >  $VALUE

logger "somfyrelais $* index=$index nieuwpct=$nieuwpct"
```

The argument of the script is the index of the device in the domoticz list. In that way, it is not necessary to search in the device list.

The function `sw` emulates a short push on a button.

The position of the slider is read using

```
nieuwpct=$(curl -s "http://$DOMOTICZ/json.htm?type=devices&rid=${index}" | jq -r '.result[0].Level')
```

5. Some simple devices

5.1 Being home

For some scripts it can be important whether I am home. For example: certain lights will go on only when I am there, otherwisw they stat out.

My pressence at home is determined by the fact that my phone is there. My DHCP server hands out a fixed address to the phone, based on the MAC-ID. On th phone, I set the sleep policy to "never":

```
Settingss
    -> Wireless & Network Setting
        -> Wifi Settings
            -> Press Menu button
                -> Advanced
                    -> WiFi Sleep Policy
```

That was enough for my Android phone to respond to pings.

I made a virtual hardware device type "Dummy (Does nothing, use for virual switches only)". For that hardware, I made a virtual sensor, type switch. The name of that sensor is assigned to the variable `DOMO_DEV` in the script below.

The script polls the phone and sets the value of the virtual switch.

```
#!/bin/bash
#REMOTE@ domoticz.home /usr/local/bin/ljthuis
########################################################################
# Set or change the hostnames below to match your environment
########################################################################

DOMOTICZ='domoticz.home:8888'
TARGET=192.168.178.219
DOMO_DEV='laurent-jan_thuis'

########################################################################
helpme(){
cat <<EOF

NAME: Am i at home?

EOF
}

if [ "$1" = "-h" ] ; then
          helpme
          exit 0
fi

verbose=0
if [ "$1" = "-v" ] ; then
          verbose=1
fi

debug(){
          if [ "$verbose" = 1 ] ; then
                    echo $*
          fi
}


tmp=$(mktemp)
now=$(date)

# Get list of devices
curl --silent  "http://$DOMOTICZ/json.htm?type=command&param=devices_list" |
sed 's/"//g;s/,//' |
while read nv col val ; do
          if [ "$nv" = 'name' ] ; then
```

```
                    name="$val"
          elif [ "$nv" = 'value' ] ; then
                    echo "$val $name"
          fi
done > $tmp


val=0
if ping -c5 $TARGET > /tmp/last_ljm.out 2> /tmp/last_ljm.err ; then
          val=1
fi
debug "Value: $val"
idx=$(sed -n "s/ $DOMO_DEV//p" $tmp)
debug "idx=$idx"
if [ "idx" = "" ] ; then
          true
else
          debug curl --silent  "http://$DOMOTICZ/json.htm?type=command&param=udevice&idx=$idx&nvalue=$val"
          curl --silent  "http://$DOMOTICZ/json.htm?type=command&param=udevice&idx=$idx&nvalue=$val" > /dev/null
fi

rm -f $tmp
```

The script runs every 5 minutes from `cron`

```
#USER root@domoticz.home                                                                                    #--ljthuis.cron:root
#INSTALLEDFROM verlaine:src/domoticz                                                                        #--ljthuis.cron:root
#  m           h  D   M  dow  cmd                                                                           #--ljthuis.cron:root
   1,16,31,46  *  *   *   *   /bin/bash /usr/local/bin/ljthuis > /tmp/last_ljthuis.out 2>/tmp/last_ljthuis.err  #--ljthuis.cron:root
   6,21,36,51  *  *   *   *   /bin/bash /usr/local/bin/ljthuis > /tmp/last_ljthuis.out 2>/tmp/last_ljthuis.err  #--ljthuis.cron:root
   11,26,41,56 *  *   *   *   /bin/bash /usr/local/bin/ljthuis > /tmp/last_ljthuis.out 2>/tmp/last_ljthuis.err  #--ljthuis.cron:root
```

5.2 Internet connection

I had the impression that my Internet connection was unreliable. To get some data about how unreliable the connection was, I created a custom sensor, with the sensor unit "msec". The value of the sensor is the round-trip time for a ping to 8.8.8.8. Again, an external script updates the value of the sensor.

```
#!/bin/bash
#REMOTE@ domoticz.home /usr/local/bin/internetup
#########################################################################
# Set or change the hostnames below to match your environment
#########################################################################

DOMOTICZ='domoticz.home:8888'
TARGET=8.8.8.8

#########################################################################
helpme(){
cat <<EOF

NAME:
          internetup - test if internet is up

SYNOPSIS:
          p1_report

DESCRIPTION:

Internetup pings $TARGET to see if the Internet is reachable.

The default URL for Domoticz is '$DOMOTICZ' and the default
target for ping is $TARGET

Internetup is typicaly started by cron to ensure a regular data feed to
Domoticz.

EOF
}

if [ "$1" = "-h" ] ; then
          helpme
          exit 0
fi

verbose=0
if [ "$1" = "-v" ] ; then
          verbose=1
fi
```

```
debug(){
          if [ "$verbose" = 1 ] ; then
                    echo $*
          fi
}


tmp=$(mktemp)
pingout=$(mktemp)
now=$(date)

val=500
if ! ping -c1 8.8.8.8 > $pingout ; then
          sleep 1
          ping -c1 8.8.8.8 > $pingout
fi
if grep -q 'time=' $pingout ; then
          val=$(sed -n 's/ ms//; s/.*time=//p' $pingout)
fi
debug "Value: $val"

# Get list of devices
curl --silent  "http://$DOMOTICZ/json.htm?type=command&param=devices_list" |
sed 's/"//g;s/,//' |
while read nv col val ; do
          if [ "$nv" = 'name' ] ; then
                    name="$val"
          elif [ "$nv" = 'value' ] ; then
                    echo "$val $name"
          fi
done > $tmp

inet_idx=$(sed -n 's/ internet_up//p' $tmp)

debug "inet_idx=$inet_idx"

if [ "inet_idx" = "" ] ; then
          echo "$now No inet dev">> /tmp/last_inet
else
          debug curl --silent  "http://$DOMOTICZ/json.htm?type=command&param=udevice&idx=$inet_idx&nvalue=1&svalue=$val"
          curl --silent  "http://$DOMOTICZ/json.htm?type=command&param=udevice&idx=$inet_idx&nvalue=1&svalue=$val" > /dev/null
          echo "$now Internet: $val" >> /tmp/last_inet
fi

tail -1024 /tmp/last_inet > $tmp
mv $tmp /tmp/last_inet

rm -f $tmp $pingout
```

And again, launched from cron:

```
#USER root@domoticz.home                                                                                #--internet_up.cron:root
#INSTALLEDFROM verlaine:src/domoticz                                                                    #--internet_up.cron:root
#  m           h  D   M  dow  cmd                                                                       #--internet_up.cron:root
   0,15,30,45  *  *   *   *   /bin/bash /usr/local/bin/internetup > /tmp/last_internetup.out 2>/tmp/last_internetup.err  #--internet_up.cron:root
   5,20,35,50  *  *   *   *   /bin/bash /usr/local/bin/internetup > /tmp/last_internetup.out 2>/tmp/last_internetup.err  #--internet_up.cron:root
   10,25,40,55 *  *   *   *   /bin/bash /usr/local/bin/internetup > /tmp/last_internetup.out 2>/tmp/last_internetup.err  #--internet_up.cron:root
```

The results for my Ziggo line where, that I had a down time of more than 15 minutes about every two days. Also, the tound-trip time varied wildly, between 8msec and 100msec. That jitter was a serious problem in video calls.

Contents