SwitchHub 0.5

Sunday, April 10, 2016

Table of Contents

1 Introduction

1.1 SwitchHub

SwitchHub is a command line application for computers running a Linux based operating system that allows flexible control of the Telldus Tellstick[1]. The Telldus Tellstick is a device that can be controlled from a computer to control power switches remotely, refer to Illustration 1. SwitchHub can be adopted to control similar devices that can be controlled from the command line.

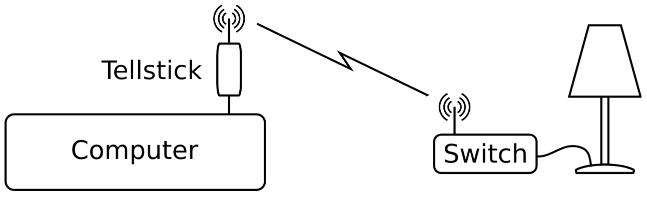


Illustration 1: Controlling an electrical device

The scheduling of events (i.e. scheduling switches to turn on or off, or dim) is made by creating Boolean expressions[2], one per switch. This is an example of a Boolean expression used as an event definition:

```
on = 06:15 <= t < 22:15 and weekday
```

When the Boolean expression becomes True (between 06:15 and 22:15 during any weekday), the switch is turned on. When the Boolean expression becomes False (outside the time interval or on a weekend), the switch is turned off. To write Boolean expressions, you will use:

- Variables, such as for example 't' or "weekday"
- Boolean operators, "and", "or" and "not"
- Relational operators, for example '<' and ">=".

1.2 SwitchHub characteristics

- SwitchHub can use plug-ins. The core functionality (based on Python standard libraries) is implemented in the SwitchHub software itself. You then have the possibility to add plug-ins to increase the number of variables that can be used in the event definitions. Some plug-ins are bundled with SwitchHub.
- SwitchHub can control both switches and dimmers.
- The scheduling of events is very flexible. This is because:
  - You have a high degree of freedom when you write the Boolean expressions.
  - The amount of meaningful variables.
  - You can add variables by writing your own plug-ins, or running the plug-ins that follow with SwitchHub.
- SwitchHub is made to run unattended in the background. Once configured and started, SwitchHub will be out of your way.

1 Tellstick is a product made by Telldus Technologies, www.telldus.se.
2 https://en.wikipedia.org/wiki/Boolean_expression

- SwitchHub does not have a GUI. You will edit a few text files to configure the program and the events.
- SwitchHub is written in Python 3. This is only of interest if you want to read the code or make changes to it.

1.3 The SwitchHub project

The program files and this document resides at GitHub at www.github.com/switchhub. You are encouraged to provide feedback in any form, bug reports, feature requests or code.

Besides maintaining SwitchHub for the intended usage with the characteristics as described in this document, the following goals are also considered important:

- Pythonic code (that is readable code, written the" Python way").
- Understandable and sufficient documentation.

1.4 This document

This document assumes that you have a general understanding of computers, especially computers that run a variant of the GNU/Linux operating system.

This document describes SwitchHub as it works in its current version. If you want to now what changes and new features to expect in the future, refer to https://github.com/thoelf/switchhub/issues.

1.5 Legal notes

SwitchHub is released as free/libre open source software under the GNU General Public License v3. For the license text, refer to the `LICENSE` file or to www.gnu.org/licenses/gpl.html.

You will use SwitchHub (the program) at your own risk. The program is not intended for mission critical tasks, where the condition of physical property or the life or well-being of humans, animals or other living creatures depends on how the program behaves.

1.6 Abbreviations and acronyms

For a list of abbreviations and acronyms used in this document, refer to table.

Abbreviation/acronym	Description
GNU	GNU is Not Unix
GUI	Graphical User Interface
IP	Internet Protocol
LAN	Local Area Network
JSON	JavaScript Object Notation
MAC	Media Access Control
URL	Uniform Resource Locator
UTC	Coordinated Universal Time

2 Configuration files

2.1 Introduction

SwitchHub uses two configuration files which are installed in the directory /etc/switchhub:

- events, for the event definitions.
- switchhub, to control certain aspects of the program execution.

SwitchHub will read the configuration files when it starts. If you want SwitchHub to be aware of changes you make in a configuration file when SwitchHub is running, you must restart SwitchHub.

The plug-ins does not have any configuration files. You might have to edit the settings inside the plug-in files themselves.

2.2 The event definitions file, events

2.2.1 Structure

The event defintion file (/etc/switchhub/events) can have for example this contents:

```
[Kitchen lamp]
id = 1
on = 06:00 < t <= 12:30 and monday

[Bedroom lamp]
id = 2
dim_10 = 06:00 < t <= 07:00 and weekday
dim_90 = 15:00 < t <= 22:00 and ping_sven
```

The event defintion file contains:

- The device to control, for example “Kitchen lamp” in square brackets.
- A device id number.
- The event type: on, only_on, only_off or dim_<xyz>.
- An event expression with variables, Boolean operators and relational operators.

The device id number must match the id in /etc/tellstick.conf.

When using the event type “on”, the device will be powered on when the expression becomes True and turned off when the expression becomes False.

When using the event type “only_on”, SwitchHub will power on the device when the expression becomes True, but will not power off the device when the expression becomes False. The opposite is true for the event type “only_off”.

The event type “dim_<xyz>” (e.g. dim_50) can be used together with dimmers. You can also control a dimmer with the other event types. You can mix event types for a device, but you cannot use identical event types for a device, such as two occurrences of “dim_50”.

Dimmers behaves differently. When you use dimmers, you must be aware of the different dimlevels to send to the device. You can experiment with different dimlevels with this command (requires that you have installed telldusd):

```
tdtool --dimlevel <level> --dim <device id number>
```

You might find that for your dimmer, the max output corresponds to a dimlevel of for example 210.

2.2.2 The built-in variable t

The only built-in variable is t which corresponds to the current time in the format HH:MM, for example 22:45.

2.2.3 Boolean operators

For the Boolean operators that you can use when you define events, refer to Table 1.

Boolean operator	Description
and	This operator is used when you need to express that two variables or expressions must be True for the expression to be True. The expression A and B is True only if both A and B is True. (A = True) and (B = True) => True (A = True) and (B = False) => False (A = False) and (B = True) => False (A = False) and (B = False) => False
or	This operator is used when you need to express that any variable or expression must be True for the expression to be True. The expression A or B is True only if both A and B is True or if either A or B is True. (A = True) or (B = True) => True (A = True) or (B = False) => True (A = False) or (B = True) => True (A = False) or (B = False) => False
not	This operator is used when you want to express that a variable or expression must be False for the expression to be True. not A is only True is A is False. not (A = True) => False not (A = False) => True

Table 1: Boolean operators

2.2.4 Relational operators

For the relational operators that can be used when defining events, refer to Table 2.

Relational operator	Description
==	"Equal to" `5 == 5` is True

Relational operator	Description
	`5 == 6` is False
`!=`	"Not equal to" `5 != 6` is True `5 != 5` is False
`<`	"Less than" `5 < 6` is True `6 < 5` is False
`>`	"Greater than" `6 > 5` is True `5 > 6` is False
`>=`	"Greater than or equal to" `6 >= 5` is True `5 >= 5` is True `5 >= 6` is False
`<=`	"Less than or equal to" `5 <= 6` is True `5 <= 5` is True `6 <= 5` is False

Table 2: Relational operators

2.2.5 Example 1 – on and time

```
[aquarium]
id = 1
on = 14:15 <= t < 22:15
```

The device named ”aquarium” with id number '1' will be on between 14:15 and 22:15 (because that is when the expression is True).

2.2.6 Example 2 – only_on

```
[kitchen_lamp]
id = 2
only_on = t == 06:15 and sunday
```

”kitchen_lamp” will be turned on at 06:15 on Sundays. Note the difference between ”on” in Example 1 and ”only_on” in this example. ”only_on” is used to turn a device on, but the device will not be turned off when the expression becomes False. ”only_on” is useful when you want to turn off the device manually.

2.2.7 Example 3 – only_off

```
[tv]
id = 3
```

```
only_off = t == 03:00
```

”tv” will be turned off at 03:00. The device can only be turned on manually.

2.2.8 Example 4 – multiple rows

```
[bedroom_lamp]
id = 5
on =  (06:00 <= t < 07:15 and weekday) or \
      (16:20 <= t < 22:30 and weekday) or \
      (15:00 <= t < 22:30 and not weekday)
```

This example shows how it is possible to build long expressions while keeping the readability. Each row contains an expression that, if True will make the expression True.

2.3 The program configuration file, switchhub

2.3.1 Introduction

The file /etc/switchhub/switchhub contains the following settings when installed.

The settings section:

Setting	Value	Description
`log_level`	`WARNING`	The log level. One of DEBUG, INFO, WARNING, ERROR, CRITICAL. The log level DEBUG will make all types of log messages to be written to the log file. If you choose for example the log level WARNING, only log messages of the type WARNING, ERROR and CRITICAL will be written to the log file (/var/log/switchhub). The log file is rotated. Refer to /etc/logrotate.d/switchhub. The logging functionality is not fully implemented, so many fail states will not be logged
`event_config`	`/etc/switchhub/`	The directory for the event configuration file. It must end with a trailing '/'. The file is read when SwitchHub starts, and only then
`repeats`	`1`	With this setting you can make SwitchHub repeat its calls to the devices you are controlling. You might want to do repeated transmitts if the connection between the Tellstick and the receiving switches is poor
`receive_buf`	`4096`	The size of the receiving buffer. You might have to increase the buffer size if your plug-ins sends a lot of data
`port`	`8001`	The port on which SwitchHub will accept connections from plug-ins. Make sure that only trusted applications can access the port
`Wait for`	`12`	When SwitchHub starts, it will wait this time until it

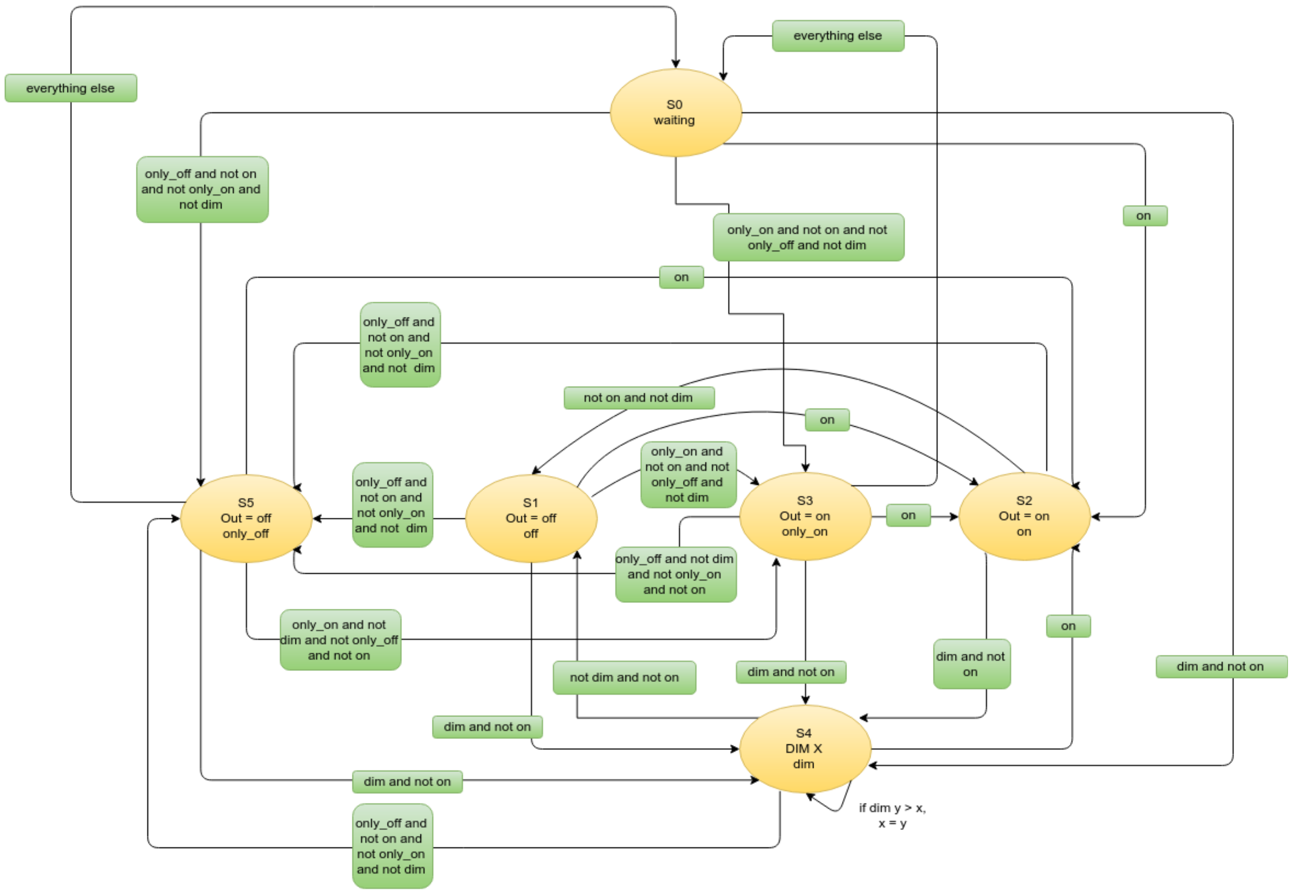
Setting	Value	Description
`plugins`		evaluates the expressions. This allows for the plugins to get data from for example the internet

The plug-ins section:

Setting	**Value**	**Description**
`<name of plugin 1>`	`<path to plugin 1>`	The plug-ins listed here will be started by SwitchHub when it starts
`<name of plugin 2>`	`<path to plugin 2>`	
`...`	`...`	

3 State diagram for the control of the switches

Refer to Illustration 2 for an explaination of the inner workings of the control of the switches. This is for your information only.



S0
waiting
everything else
everything else
only_off and not on and not only_on and not dim
only_on and not on and not only_off and not dim
on
on
only_off and not on and not only_on and not dim
not on and not dim
on
only_on and not on and not only_off and not dim
S5
Out = off
only_off
only_off and not on and not only_on and not dim
S1
Out = off
off
S3
Out = on
only_on
on
S2
Out = on
on
only_off and not dim and not only_on and not on
only_on and not dim and not only_off and not on
not dim and not on
dim and not on
dim and not on
on
dim and not on
dim and not on
S4
DIM X
dim
dim and not on
if dim y > x, x = y
only_off and not on and not only_on and not dim

Illustration 2: State diagram for the control of the switches

4 Description of the software components

4.1 Software components

4.1.1 Python 3

Python is a programming language and the software that will interpret and compile the code in the different SwitchHub modules.

4.1.2 telldusd

telldusd is the program that communicates with the Tellstick. SwitchHub calls telldusd to operate a switch or dimmer.

4.1.3 Plug-ins

These plug-ins are included in the software package from GitHub (but they are not enabled by default):

- The calendar plug-in (calendar_data.py) gives you variables for holidays.
- The sunrise/sunset plug-in (suntime.py) gives you variables for sunrise and sunset.
- The read files plug-in (readfiles.py) lets SwitchHub read variable values from files that you define.
- The sensor plug-in (sensor_receiver.py) gives you data from a motion sensor or any other sensor that can communicate with the Tellstick Duo.
- The weather plug-in (weather.py). Gives you a text string that describes the current weather.
- The ping plug-in (ping.py) gives you boolean variables for if a host is available or not.

For more information about these plug-ins and plug-ins in general, refer to chapter 7.

5 Installing SwitchHub

5.1 Introduction

This instruction is tested on Ubuntu Server 14.04 32-bit and Ubuntu Desktop 13.10 64-bit. It will probably work without major tweaks on other Ubuntu versions or other Linux distributions (especially those in the Debian/Ubuntu family). The procedure describes how to do the installation from the terminal. Most or all steps can be done from the GUI if you prefer that.

5.2 Install Python 3

1. Most Linux distributions already have Python 3 installed. To verify that you have Python 3 installed, enter this command:

   ```
   python3 --version
   ```

   The expected output is something like: `Python 3.3.2+`

2. If you are missing Python 3, install it with this command:

   ```
   sudo apt install python3
   ```

5.3 Install unzip

1. Install unzip (which will be used to unzip the downloaded SwitchHub software):

   ```
   sudo apt install unzip
   ```

5.4 Install telldusd

1. Install telldusd, refer to http://developer.telldus.com/wiki/TellStickInstallationUbuntu.
2. Configure telldusd, refer to http://developer.telldus.com/wiki/TellStick_conf.
3. Initialize a switch. Refer to the manual for the switch.
4. Test that you can turn a device on and off with these commands:

   ```
   tdtool --on <device id>
   tdtool --off <device id>
   ```

   If it is a dimmer, try this command:

   ```
   tdtool –dimlevel <level> --dim <device id>
   ```

5.5 Install SwitchHub

1. Make sure that you are in your home directory:

   ```
   cd
   ```

2. Download SwitchHub:

   ```
   wget https://github.com/thoelf/switchhub/archive/master.zip
   ```

3. Unpack the downloaded file:

   ```
   unzip master.zip
   ```

4. Go to the unpacked directory:

   ```
   cd switchhub-master
   ```

5. Run the installation script:

   ```
   sudo ./install.sh
   ```

6. Go to your home directory:

   ```
   cd
   ```

7. If you do not want to keep the downloaded file, remove it:

   ```
   rm master.zip
   ```

8. If you do not want to keep the install directory, remove it:

   ```
   rm -rf switchhub-master
   ```

5.6 Improving the robustness of SwitchHub

In the unlikely event of a program crash, SwitchHub will not restart automatically. To make SwitchHub restart automatically, you might want to add a command like this in your user's crontab to be run e.g. once every fifteen minutes:

```
/usr/bin/pgrep -f "python3 /opt/switchhub/switchhub.py" >/dev/null ||
(echo "Cron: Switchhub started" $(date) >> /var/log/switchhub.log &&
```

```
/opt/switchhub/switchhub.py)
```

6 Managing SwitchHub

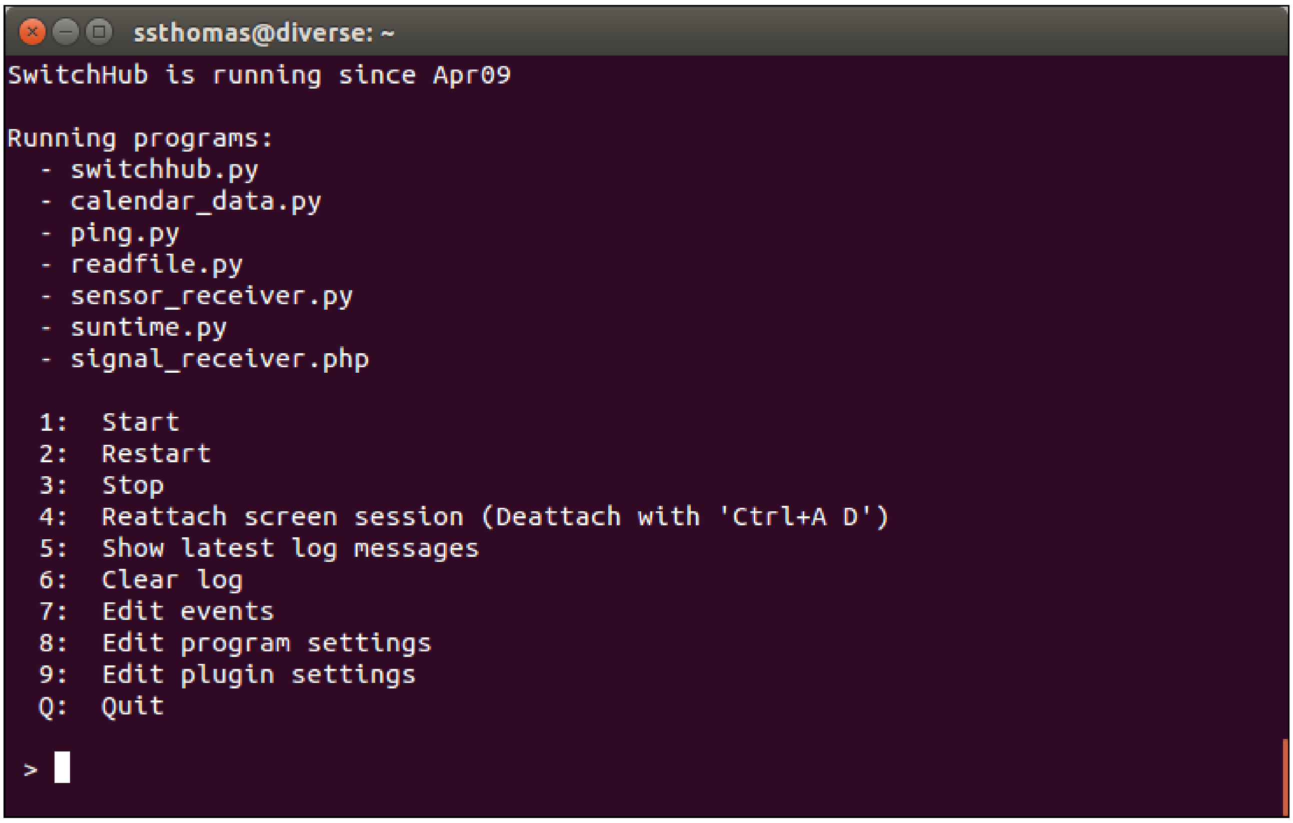
6.1 Managing SwitchHub

1. Make sure that you are in your home directory (or have the script switchhub.sh in your PATH):

   ```
   cd
   ```

2. Start the SwitchHub management consol (refer to Illustration 3):

   ```
   ./switchhub.sh
   ```



```
ssthomas@diverse: ~
SwitchHub is running since Apr09

Running programs:
  - switchhub.py
  - calendar_data.py
  - ping.py
  - readfile.py
  - sensor_receiver.py
  - suntime.py
  - signal_receiver.php

  1:  Start
  2:  Restart
  3:  Stop
  4:  Reattach screen session (Deattach with 'Ctrl+A D')
  5:  Show latest log messages
  6:  Clear log
  7:  Edit events
  8:  Edit program settings
  9:  Edit plugin settings
  Q:  Quit

 >
```

Illustration 3: SwitchHub management console

7 Plug-ins

7.1 Introduction

By adding a plug-in you can increase the number of variables known by SwitchHub. You can then use these extra variables in the event definitions. You can write your own plug-ins.

You will find the plug-ins that are included with SwitchHub in the directory /opt/switchhub/plugins.

The plug-ins that follows with SwitchHub are not activated by default. For you to be able to use the plug-ins, you must edit the file /etc/switchhub/switchhub and do any necessary settings in the plug-in file itself.

You must also set the file permissions for the plug-in so that the group switchhub have the rights to execute the file.

The plug-ins will be started automatically by SwitchHub and be killed when SwitchHub stops.

7.2 Writing your own plug-ins

The plug-ins sends data to SwitchHub through sockets on port 8001 (or any other socket that you define). For SwitchHub to be able to collect data from the plug-in, the message from the plug-in must be sent to SwitchHub on this format:

```
<plug-in name>;<variable name>;<variable value>
```

Example:

```
temperature.sh;outdoor_temp;22
```

The plug-in variables must not conflict with the names of already existing variables known to SwitchHub (i.e. an internal variable or an variable provided by a plug-in).

Make the plug-in executable for the group switchhub.

7.3 The read files plug-in, readfiles.py

The readfile plug-in reads a variable value from the first line of a file. You define the file and the corresponding variable name in readfile.py. In the file, you can also define a default value for the variable/file that will be used if the file does not exist.

7.4 The calendar plug-in, calendar_data.py

7.4.1 Variables provided by the calendar_data plug-in

The calendar plug-in provides the variables according to Table 3.

Variable	Description
`holiday_today`	The variable value depends on: • The day in the week. The variable is True on Saturdays and Sundays. • If it's holiday or not according to the downloaded calendar. Because the calendar can contain non work-free days, the program will check the file calendar_holidays to see if you have acknowledged the day as a free day. • The dates in calendar_free_days in which you can define your own free days
`holiday_tomorrow`	The variable is True if it is holiday tomorrow. The check will be done in the same way as for the variable holiday, but the check is made using tomorrow's date
`holiday_yesterday`	The variable is True if it was holiday yesterday. The check will be done in the same way as for the variable holiday, but the check is made using yesterday's date
`monday, tuesday,`	You can use a day, or a combination of days, as variables. Each vari-

Variable	Description
`wednesday, thursday, friday, saturday, sunday`	able is True during the corresponding day
`january, february, march, april, may, june, july, august, september, october, november, december`	You can use a month, or a combination of months, as variables. Each variable is True during the corresponding month
`weekday`	The variable is True between Monday and Friday
`workday`	The variable is True between Monday and Friday when there is no holiday (refer to the holiday_today variable)

Table 3: Variables provided by the calendar plug-in

7.4.2 Settings in calendar_data.py

holidays

There is a list of holidays that you can set up. The name of the list is “holidays”. When SwitchHub have found out, by reading the automatically downloaded holiday calendar from www.webcal.fi, it will check the list to see if you have acknowledged the day to be a free day. This is what a few rows in the file can look like (in Sweden):

```
nyårsdagen
trettondagsafton
långfredag
```
…

To know what days to have in calendar_holidays, refer to www.webcal.fi. The name of the days must be written exactly as at www.webcal.fi, but the names are not case sensitive.

free days

There is a list named “free_days” in which you can enter the dates when you are planning to be free from you ordinary occupation. If you for example are planning a vacation or a day off in the middle of the week, you can put the corresponding dates in the list. This is what the dates in the file can look like:

```
2016-10-22
2016-11-03:2016-12-07
```

…

The dates in the file free_days are written in the format YYYY-MM-DD. In the example, the first row contains a single date. The second row contains a range of dates.

calendar url

You must define the web address to the calendar for your country. Edit the value for the “calurl” variable in the plugin-file if you do not live in Sweden. At www.webcal.fi, look for and define a calendar (Calendar → Other file formats) for your country and time zone in JSON-format. Click **Show download URL**, then copy/paste the url to the plugin-file. Note that you will have to duplicate the '%'-character.

7.4.3 Variables from the calendar_data plug-in

The plug-in will output the following variables:

- holiday_yesterday, holiday, holiday_tomorrow
- workday_yesterday, workday, workday_tomorrow
- weekday
- monday, tuesday, wednesday, thursday, friday, saturday, sunday
- january, february, march, april, may, june, july, august, september, october, november, december

7.5 The sun data plugin, suntime.py

7.5.1 Variables provided by the sun data plug-in

This plugin provides the variables according to Table 4.

Variable	Description
`sundown`	The variable is True between 00:00 to sunrise and between sunset to 23:59, e.g.: “00:00 <= t < 07:26 or 18:01 <= t < 23:59”

`sundown_plus_10,`
`sundown_plus_20,`
`sundown_plus_30,`
`sundown_plus_40,`
`sundown_plus_50,`
`sundown_plus_60`
	The time for sundown with offset plus 10..60 minutes.
	With the sundown value above, sundown_plus_10 equals:
	“00:00 <= t < 07:36 or 18:11 <= t < 23:59”

`sundown_minus_10,`
`sundown_minus_20,`
`sundown_minus_30,`
`sundown_minus_40,`
`sundown_minus_50,`
`sundown_minus_60`
	The time for sundown with offset minus 10..60 minutes. With the sundown value above, sundown_minus_10 equals:
	“00:00 <= t < 07:16 or 17:51 <= t < 23:59”

`sunup`	The variable is True between sunrise and sunset, e.g: “07:26 <= t < 18:01”

`sunup_plus_10,`
`sunup_plus_20,`
`sunup_plus_30,`
`sunup_plus_40,`
`sunup_plus_50,`
`sunup_plus_60`
	The time for sunup with offset plus 10..60 minutes.
	With the sunup value above, sunup_plus_10 equals:
	“07:36 <= t < 18:11”

`sunup_minus_10,`
`sunup_minus_20,`
`sunup_minus_30,`
`sunup_minus_40,`
`sunup_minus_50,`
`sunup_minus_60`
	The time for sunup with offset minus 10..60 minutes.
	With the sunup value above, sunup_minus_10 equals:
	“07:16 <= t < 17:51”

Table 4: Variables provided by the suntime plug-in

7.5.2 Settings in suntime.py

Variable	Description
`utc_diff`	Your local time minus UTC time. Local time is "normal" time, without daylight saving
`lat`	The latitude and longitude of your location. Refer to for example http://www.latlong.net/
`lng`	
`default_sunup`	Times for sunrise from the middle of each month, or any other default time that you prefer
`default_sundown`	Times for sunset from the middle of each month, or any other default time that you prefer
`sundata_url`	URL for http://sunrise-sunset.org/ from where the time for sunrise and sunset is downloaded. There should be no reason to change this setting

7.6 The sensor plug-in, sensor_receiver.py

7.6.1 General

The sensor plug-in operates in cooperation with signal_receiver.php as found at... can't remember. The signal_receiver.php program is started from the sensor plug-in. Note that php must be installed in order to run signal_receiver.php. It has been tested with php5.

signal_receiver.php collects all data that the Tellstick Duo picks up and sensor_receiver.py takes care of it and knows what sensor that sent the data. You must edit sensor_receiver.py so that it detects your particular sensor. Just run signal_receiver.php from the command line and observe the strings that it outputs when a sensor sends data.

The sensor plug-in is for the moment only written for sensors that has two states (on/off) such as motion sensors and buttons.

7.6.2 Settings in sensor_receiver.py

For the settings for a bistable behaviour, refer to the table.

Setting	Description
`name`	Name of the sensor
`type`	Type, bistable or monostable. Choose “bistable” if the variable that represents the sensor output should be stable in both the on and off position. Example: A button sensors a push. The corresponding variable becomes True until the next push (when it becomes False)
`initial_state`	True or False (on or off)
`bounce_filter_time`	For how many seconds the sensor object (i.e. the soft representation of the sensor) will be immune from new input after valid input has been detected
`toggle_on_input`	If True, toggles the output variable for a input string

`str_set`	The strings from signal_receiver.php that is valid to make the sensor's output variable True
`str_reset`	The strings from signal_receiver.php that is valid to make the sensor's output variable False. Set to “None” if such strings does not exist

For the settings for a monostable behaviour, refer to the table.

Setting	Description
`name`	Name of the sensor
`type`	Type, bistable or monostable. Choose “monostable” if the variable that represents the sensor output should be stable in the off position. Example: A motion sensor detects movement. The corresponding variable becomes True and will return to its False state after a certain time (see time_to_rest)
`initial_state`	True or False (on or off)
`bounce_filter_time`	For how many seconds the sensor object (i.e. the soft representation of the sensor) will be immune from new input after valid input has been detected
`time_to_rest`	Wait for this time before returning to the stable state
`extend_time_to_rest`	True if the timer for time_to_rest can be reset when new input is detected, otherwise False
`str_set`	The strings from signal_receiver.php that is valid to make the sensor's output variable True
`str_reset`	The strings from signal_receiver.php that is valid to make the sensor's output variable False. Set to “None” if such strings does not exist

7.7 The ping plug-in, ping.py

A host on the internet or most likely on your LAN. Configure the host IP or URL in ping.py. Any number of hosts can be defined. You can use the variables from the plug-in to let an expression depend on whether you or a family member are home or not (by letting SwitchHub ping your cell phone(s)). This only works if:

- You have a LAN
- The IP addresses of the hosts are set dependent on the MAC addresses, so that the IP addresses are consistent over time.