SwitchHub 0.3

Monday, March 31, 2014

Table of Contents

1 Introduction

1.1 SwitchHub

SwitchHub is a command line application for Linux (exclusively for the time being) 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 easily adopted to control other similar devices.

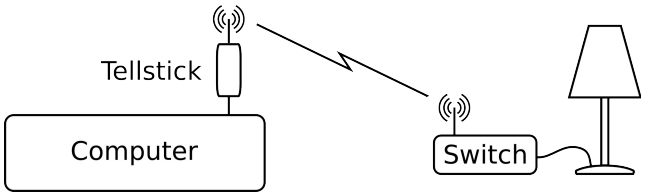

Illustration 1: Controlling an electrical device

The scheduling of events (i.e. scheduling switches to turn on or off) 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 a 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, e.g. 't' and “weekday”
- Boolean operators, “and”, “or” and “not”
- Relational operators, e.g. '<' and “>=”.

1.2 SwitchHub characteristics

For you to know what to expect from SwitchHub, here follows a list of some of the important characteristics of the application:

- SwitchHub is lightweight and will not use much of your system's resources.
- SwitchHub can use plug-ins. The core functionality (based on Python standard libraries) is implemented in the SwitchHub software itself. The user then have the possibility to add plug-ins to increase the number of variables that can be used in the event definitions.
- 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 installing the plug-ins that follow with SwitchHub.
- SwitchHub is intended to run unattended as a server daemon. Once configured and started,

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

SwitchHub will be out of your way.

- SwitchHub does not have a GUI. You will edit text files (there is two of them) 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 health of humans or animals depends on how the program behaves.

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 stop and start SwitchHub.

Depending on which plug-ins you use, there can be more configuration files to care for in /etc/switchhub/plugins/. There can also be settings in the plug-in executables.

2.2 The event definitions file, events

2.2.1 The built-in variables used by the event definitions

For descriptions of the built-in variables you can use when you write event definitions, refer to Table 1.

Variable	Description
`t`	The time right now
`weekday`	The variable is True between Monday and Friday
`monday, tuesday, wednesday, thursday, friday, saturday, sunday`	You can use a day, or a combination of days, as variables. Each variable 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
`party`	You can use this variable as a" kill switch", so that when party is True, the whole Boolean expression is True irrespective of the values of the other variables. In other words, if you have a party, you can set the party variable to True to prevent the lights to go out to early. The variable will automatically reset at the time specified in the file switchhub. In the current version of SwitchHub, this variable is always False. In the future, this document will describe a way to set and reset the variable from an Android phone using Tasker
`ping['host']`	A host on the internet or most likely on your LAN. Configure the host IP or URL in the file switchhub. Any number of hosts can be defined. You can use this variable 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 dependent on the MAC addresses, so that the IP addresses are consistent over time

Table 1: Built-in variables

2.2.2 Boolean operators

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

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 2: Boolean operators

2.2.3 Relational operators

For the relational operators that can be used when defining events, refer to Table 3.

Relational operator	Description
==	“Equal to” `5 == 5` is True `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 3: Relational operators

2.2.4 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). The device name ”aquarium” is a unique name that is intended for you to know which device it is. The id number, '1', is a unique number that is used in the calls to the Tellstick (or a similar device). If you are using Tellstick, the id number must match the id number for the device in /etc/tellstick.conf.

2.2.5 Example 2 – only_on, time and ping['host']

```
[kitchen_lamp]
id = 2
only_on = t == "06:15" and ping['t_s5']
```

”kitchen_lamp” will be turned on at 06:15 if t_s5 is present on the LAN. Note the difference between ”on” in Example 1 and ”only_on” in Example 2. ”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.

2.2.6 Example 3 – only_off and time

```
[tv]
id = 3
only_off = t == "03:00"
```

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

2.2.7 Example 4 – multiple rows and multiple variables

```
[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) or \
      party
```

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 whole expression True.

2.3 The program configuration file, switchhub

2.3.1 Introduction

Each of the following sub chapters corresponds to a section in the file /etc/switchhub/switchhub. The shown settings are the default settings.

2.3.2 The logging section

```
[logging]
log_level = WARNING
```

The log level defines what kind of log info you will get in the log file /var/log/switchhub.log. You can choose from the log levels that follow:

- 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.

The log file is rotated. Refer to switchhub_logrotate for information about the configuration.

2.3.3 The misc section

```
[misc]
event_config = /etc/switchhub/
```

The path to the event configuration file events. With this setting you can store the event configuration file in a directory that is synchronized with a cloud storage service. In that way you will be able to control SwitchHub from any location in the world. This is not implemented in the current version of SwitchHub (the file is only read when SwitchHub starts).

```
party_ends = 02:00
```

The party variable will be set to False at this time. The party variable is always False in the current version of SwitchHub.

```
repeats = 4
```

With this setting you can make SwitchHub repeat its calls to the devices you are controlling. You might want to do repeated calls if the connection between the Tellstick and the receiving switches is poor.

2.3.4 The ping_ip section

```
[ping_ip]
thomas = 192.168.1.110
marlene = 192.168.1.111
```

This is where you define the hosts that SwitchHub will ping, to know if they are on the LAN or not. You can also specify an URL here. The variable name for the ping to a host is on the form ping['host'].

2.3.5 The timer section

```
[timer]
ping_off_delay = 10
```

Time in minutes for SwitchHub to notice that the contact to a host (that is configured in the [ping_ip] block) is lost. In other words, the time in minutes for ping['host'] to go from True to False when contact with the host is lost. This is a global setting in the current version of SwitchHub and will therefore effect ping to all hosts. The off_delay is good to have if you want a lamp to go off some time after you leave the house. The off_delay also functions as a filter, so that if you loose the wifi-connection now and then, the devices that you control will not be affected.

```
turn_around = 60
```

The time in seconds for how often SwitchHub will check the value of the Boolean expressions and possibly change the states of the switches. This setting is not implemented in the current version of SwitchHub, so the turn around time is always 60 seconds.

3 Description of the software components

3.1 Software components

3.1.1 Python 3

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

3.1.2 telldusd

telldusd is the program that communicates with the Tellstick. SwitchHub calls telldusd when it is time to operate a switch.

3.1.3 SwitchHub

SwitchHub consists of 4 modules:

- switchhub.py is the main module. This is the program that you run when you want to start SwitchHub.
- get_plugin_data.py is the module that collects data from any plug-ins.
- operate_switch.py – This module tells the telldusd software to turn a switch on or off.

3.1.4 Plug-ins

The Google Calendar plug-in (gcalendar.py) is included, but will not be enabled by default. For more information about this plug-in and plug-ins in general, refer to chapter 6.

4 Installing SwitchHub

4.1 Introduction

This instruction is tested on Ubuntu Server 13.10, 32-bit and Ubuntu Desktop 13.10, 64-bit. It will probably work without major tweaks on other Ubuntu versions or other Linux distributions. The procedure contains how to do the installation from the terminal. Most or all steps can be done from the GUI if you prefer that.

4.2 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-get install python3
   ```

4.3 Install unzip

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

   ```
   sudo apt-get install unzip
   ```

4.4 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>
   ```

4.5 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. You might want to add /opt/switchhub to your path or make soft links in your home directory. Follow these steps to make soft links in your home directory:

   ```
   ln -s /opt/switchhub/switchhhub_start switchhub_start
   ln -s /opt/switchhub/switchhhub_status switchhub_status
   ln -s /opt/switchhub/switchhhub_stop switchhub_stop
   ```

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

   ```
   rm master.zip
   ```

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

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

5 Managing SwitchHub

5.1 Starting SwitchHub

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

   ```
   cd
   ```

2. Start SwitchHub:

   ```
   ./switchhub_start
   ```

3. Detach from the screen session:

   ```
   Ctrl+A D
   ```

   If you want automatic detach, edit switchhub_start according to the instructions in the file.

5.2 Stopping SwitchHub

1. Make sure that you are in your home directory:
   ```
   cd
   ```
2. Stop SwitchHub:
   ```
   ./switchhub_stop
   ```

5.3 Checking the status of SwitchHub

1. Make sure that you are in your home directory:
   ```
   cd
   ```
2. Check if SwitchHub is running and see the 10 latest log messages (if any):
   ```
   ./switchhub_status
   ```

6 Plug-ins

6.1 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 included plug-ins are:

- gcalendar.py
- wunderground_weather.py

The plug-ins above are not activated by default. For you to be able to use the plug-ins, you must follow the install procedures in chapter 7 and 8.

6.2 Activating a plug-in

To activate a plug-in you have to copy it to one of the directories that follow in /opt/switchhub/plugins:

- 1_minute
- 15_minutes
- hour
- day.

The name of a directory reflects how often the plug-ins in the directory will be executed and its data be collected by SwitchHub. All plug-ins will be executed once when SwitchHub starts. The plug-ins in 1_minute/ will then be executed every full minute. The plug-ins in 15_minutes/ will be executed every full quarter. The plug-ins in hour/ will be executed every full hour. The plug-ins in day/ will be executed every day at 00:00.

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

6.3 Writing your own plug-ins

The plug-ins communicates to SwitchHub through stdout. The message from the plug-in must be on this format:

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

Example (output on stdout):

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

If the plug-in requires a configuration file, you are recommended to put the configuration file in the directory /etc/switchhub/plugins. Give the configuration file the same name as the plug-in executable, but exclude the file extension.

To let the plug-in output more than one variable, output the strings on subsequent rows.

Example (output of more than one variable on stdout):

```
outdoor_sensor.sh;outdoor_temp;22
outdoor_sensor.sh;outdoor_hum;47
```

The plug-in variables must not conflict with the names of already existing variables known to SwitchHub. SwitchHub will inform you about any name conflicts.

Make the plug-in executable for the group switchhub.

You must not make a plug-in run forever. A well behaved plug-in should just fetch its data, output the data on stdout and then exit.

7 The Google Calendar plug-in, gcalendar.py

7.1 Variables provided by gcalendar.py

The Google Calendar plug-in provides the variables according to table Table 4.

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 Google Calendar. Because Google Calendar can contain non work-free days, the program will check the file gcalendar_holidays to see if you have acknowledged the day as a free day. • The dates in gcalendar_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
sundown	The variable is True between 00:00 to sunrise and between sunset to 23:59
sunup	The variable is True between sunrise and sunset
workday	The variable is True between Monday and Friday when there is no holiday (refer to the holiday_today variable)

Table 4: Variables provided by the Google Calendar plug-in

7.2 Configuration files for the Google Calendar plug-in

7.2.1 The holidays configuration file, gcalendar_holidays

The file /etc/switchhub/plugins/gcalendar_holidays is a list of holidays that you have acknowledged as free days. When SwitchHub have found out, by reading Google's holiday calendar, that it might be a holiday, it will check the gcalendar_holidays to see if you have acknowledged the day to be a free day. This procedure is necessary because not all days in Google's holiday calendar are free days. This is what a few rows in the file can look like (in Sweden):

```
Nyårsdagen
Trettondagsafton
Långfredag
…
```

To know what days to have in gcalendar_holidays, follow the instructions on https://support.google.com/calendar/answer/37098?hl=en to add (or preview) the holiday calendar for your country. You will have to browse through a whole year to find the days to add to gcalendar_holidays. The name of the days must be written exactly as in Google Calendar, but the names are not case sensitive.

The contents of gcalendar_holidays will effect the variables holiday_yesterday, holiday_today and holiday_tomorrow.

7.2.2 The free days configuration file, gcalendar_free_days

In the file /etc/opt/switchhub/plugins/gcalendar_free_days you can add 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 gcalendar_free_days. This is what the dates in the file can look like:

```
2014-10-22
2014-11-03:2014-11-07
```

...

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

The contents of the file gcalendar_free_days will affect the variables holiday_yesterday, holiday_today and holiday_tomorrow.

7.3 Settings in gcalendar.py

In the file gcalendar.py you can define the names of the holidays calendar and the sunrise/sunset calendar. Change this setting if you do not live in Sweden:

```
holidays_calendar = "Helgdagar i Sverige"
```

Change this setting if you do not live in Linköping:

```
sun_calendar = "Soluppgång och solnedgång för Linköping"
```

Refer to your Google Calendar to know what calendars to use.

7.4 Software components

7.4.1 googlecl

googlecl is a command line tools for the Google Data APIs. googlecl is used by SwitchHub to access data from Google Calendar. The use of googlecl requires that you have a Google account and that you approve googlecl to access the Google account.

7.4.2 gdata-python-client

“The Google Data Python Client Library provides a library and source code that make it easy to access data through Google Data APIs.” (http://code.google.com/p/gdata-python-client/)

7.5 Installing the Google Calendar plugin

7.5.1 Install googlecl

1. Download and install googlecl from https://code.google.com/p/googlecl/downloads/list. SwitchHub has been tested with googlecl version 0.9.14-2. You might also find googlecl in your repositories, but it could be an older version, not guaranteed to work as expected.
2. To approve googlecl's access to Google Calendar, refer to chapter 7.5.3 or 7.5.4.

7.5.2 Install gdata-python-client

1. Download gdata-python-client:

   ```
   wget http://gdata-python-client.googlecode.com/files/gdata-2.0.18.tar.gz
   ```

2. Unpack the downloaded file:

   ```
   tar -xzf gdata-2.0.18.tar.gz
   ```

3. Go to the unpacked directory:

   ```
   cd gdata-2.0.18
   ```

4. Install the gdata-python-client:

   ```
   sudo python setup.py install
   ```

7.5.3 Approving googlecl to connect to Google Calendar if you have a GUI

2. Try to access Google Calendar with this command:

   ```
   google calendar list --date today
   ```

3. You will be prompted to "Please specify user:". Enter your user name on Google (i.e. your Gmail address, e.g. evert_xyz.svensson@gmail.com). Your default browser opens. If it does not open, open a browser and browse to the URL provided in the terminal.

4. Accept the request for access.

7.5.4 Approving googlecl to connect to Google Calendar if you do not have a GUI

1. Make sure that you have another computer with a GUI running and that you have access to for example Firefox or Chromium. (The computer with a GUI is not needed if you use a web browser for the command line that supports Javascript.)

2. Try to access Google Calendar with this command:

   ```
   google calendar list --date today
   ```

5. You will be prompted to "Please specify user:". Enter your user name on Google (i.e. your Gmail address, e.g. evert_xyz.svensson@gmail.com).

6. You will see a URL in the terminal window. Copy the URL to another computer and browse to the URL using for example Firefox or Chromium, then accept the request for access.

7. Return to the terminal window and push **Enter**.

7.5.5 Install the file gcalendar.py

1. Go to the plug-ins directory:

   ```
   cd /opt/switchhub/plugins
   ```

2. Move the plug-in to the day/ plug-in directory:

   ```
   mv gcalendar.py day
   ```

3. Restart SwitchHub:

   ```
   cd ..
   ./switchhub_stop
   ./switchhub_start
   ```

8 The Wunderground plug-in, wunderground.py

The Wunderground plug-in gives you the variables outdoor_temp and TBD.

To install and configure the Wunderground plug-in and its dependencies, refer to the procedure that follows (in a later version).

This plug-in is planned, but not yet implemented.