Weather23k

Index

1 Intention

This project “weather23k“ I developed since a couple of years.

I prefer running linux on my computers so I can‘t use the program “heavyweather” that was included in the WS2300 package I bought. I needed to write my on application to get the weather data to my homepage.

I searched thru the internet and found the “open2300“ library what was the solution I needed.
I implemented a small application around it.

First I ran it on a PC but this consumed a lot of power when running 7/24. So I switched to a small Alix PC.
When the Raspberry Pi was announced I decided this would be the perfect solution and I got some of these.

But the compilation of “open2300” library was very slow on the RaspPi. Therefore I decided to develop a completely new application. It should use the basics of “open2300” but not include it‘s powerful gadgets. And it will be done for linux only!

So “weather23k“ has following features to keep it small and simple :

- read data out of a WS2300 compatible weather station
- push current data as text to a web server
- push data to a log file on a web server
- configurable via a config file
- no GUI
- linux only
- C coded
- no framework needed
- code as small as possible
- open source

2 The Program weather23k

The program weather23k does

- read the data from a WS2300 compatible weather station every minute
- add the data read to a logfile on a ftp server every minute
- create a new logfile on the ftp server every day
- create a text file on the ftp server containing HTML code to show the current read data inside a web page every minute

2.1 Connection to the weather station

The computer running weather23k is connected to the weather station with either

- a serial cable (RS232) with at least connecting the lines
    - Gnd
    - RxD
    - TxD
    - DTR
    - RTS

or

- an USB to serial adapter featuring the same lines as above.

2.2 Log File

The log file is a text file named

```
yyyy_mm_dddata.log
```

where

- yyyy is the year
- mm is the month
- dd is the day

There is a line for every minute at which data is logged.

```
00:23:01   1.6 1001.8 1016.0  80 202.5 SSW  0.0   0.0   0.0  0  -1.5   1.6   0.0   0.0
```

The data are organized in columns separated by one or more blanks :

- time stamp
- temperature [°C]
- absolute pressure [hPa]
- relative pressure [hPa]
- relative humidity [%]
- wind direction [°]
- wind direction [text]
- wind speed [m/sec]
- wind speed [km/h]
- wind speed [kn]
- wind speed [bft]
- dew-point [°C]
- windchill [°C]
- rain per hour [l]
- rain per day [l]

2.3 HTML File

This file will be created every minute on the ftp server. It contains the HTML code give in the configuration file in the section [Template] with the variables filled with the values last read from the weather station. It can be included in a web page just as is.

3 Installation

3.1 Prerequisites

To get this program compiled and linked you need some programs and libraries on your computer. Sometimes these are preinstalled, sometimes not. What is missing you can get with the package manager of your preferred linux system.

```
# pacman -Syu
# pacman -S gcc
# pacman -S make
# pacman -S curl curl-debug
```

Arch Linux

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install make
$ sudo apt-get install gcc
$ sudo apt-get install curl
$ sudo apt-get install libcurl4-openssl-dev
```

Debian Linux

I found the curl and curl development libraries have very different names in the different Linux distributions. So you have to look in the internet to find witch name is the right for your preferred distribution. Take the above lines as example only!

3.2 Get the program

The program itself is available as source code only. You have to get it from GitHub by cloning it :

```
git clone https://github.com/KlabautermannSW/weather23k.git
```

or download the source as a .zip file :

```
wget https://github.com/KlabautermannSW/weather23k/archive/master.zip
```

Both will result in a directory named `weather23k` with some subdirectories.

In this directory run

```
mkdir bin
make
```

This will start compiling and linking all necessary files together and results in a binary in the subdirectory `bin`.

4 Starting the Program

This description is for systemd based linux.

You have to do the following steps to get weather23k starting on boot time and every time it stops accidentally.

Create a file named weather23k.service in /usr/lib/systemd/system :

```
#################################################################
#
# weather23k.service
# systemd service: start weather23k at start
#
#################################################################

[Unit]
Description=Start weather23k

[Service]
# Type=simple : default, necessary when using "Restart"
Type=simple
ExecStart=/home/<path>/weather23k/bin/weather23k →
/home/<path>/weather23k/conf/weather23k.conf
# Restart=always : starts the program at once if it was finished in any way !!!
Restart=always

[Install]
WantedBy=multi-user.target

# EOF
```

Then you have to set the file’s flags :

```
# chmod 644 /usr/lib/systemd/system/weather23k.service
# ls -l /usr/lib/systemd/system/weather23k.service
-rw-r--r-- 1 root root 351 30. Apr 19:56 →
/usr/lib/systemd/system/weather23k.service
```

Next you will activate the service :

```
# systemctl enable weather23k.service
Created symlink /etc/systemd/system/multi-user.target.wants/weather23k.service
→ /usr/lib/systemd/system/weather23k.service.
```

And at last start the service :

```
# systemctl start weather23k.service
```

5 Hardware

I use this program on a Raspberry Pi 1 Model B.
The weather station WS2307 is connected to the RasbPi via an USB to serial converter.
The connection to the internet is done via cable bound Ethernet.

For more information visit the internet pages :
http://www.ur9.de/infos.shtml and http://www.ur9.de/ws2300.shtml

6 Configuration file

The configuration file has several sections that are described in the next paragraphs.
A line beginning with the “#” character is interpreted as a single line comment.

6.1 [FTP]

This describes the handling and use of the ftp server.

```
server =
user =
key =
file =
logpath =
```

All data are encoded using the encode fucntion.

6.1.1 server

This is the server string as used for the ftp server connection. Example:

```
myspace.example.com
```

6.1.2 user

The ftp account’s name.

6.1.3 key

The ftp account’s password.

6.1.4 file

A text file to be included into the web page. This needs complete path and filename starting from the root directory of your ftp server. Example:

```
/root/wheather/data.html
```

6.1.5 logpath

The path where the daily logfiles will be stored on the ftp server. This must start at the root directory of your ftp server and end with a slash. Example:

```
/root/wheather/
```

6.2 [File]

```
# if no logpath is given log will saved in the current directory
# logpath =
```

6.2.1 logpath

The path where the daily logfiles will be stored on the local computer. Default is the current directory.

6.3 [Port]

Example:

```
port = /dev/ttyUSB0
# port = /dev/ttyAMA0
```

6.3.1 port

The port at which the weather station is connected. The second line is the first serial port of a Raspberry Pi.

6.4 [Template]

The template key has to be the last key!
All text following this key will be put character by character into the file “data.txt” on the ftp server. The only exception is the string "<*var=_name_*>" wherein _name_ is described in the following list. This string will be changed to the value of the given variable or to an empty string if _name_ is unknown.

temp	temperature
press	relative air pressure
hum	relative air humidity
winddir	wind direction
speed_m	wind speed [m/sec]
speed_kmh	wind speed [kmh]
speed_kn	wind speed [kn]
speed_bf	wind speed [bft]
dew	dew-point temperature
chill	windchill temperature
rph	rain per hour
rpd	rain per day
dirstr	wind direction as text
time	current time stamp

```
<table border="0" cellspacing="0" cellpadding="0">
      <tr>
            <td width="200">Messzeit</td>
            <td width="120"><*var=time*> Uhr</td>
      </tr>
      <tr>
            <td>Temperatur</td>
            <td><*var=temp*> &#176;C</td>
      </tr>
      <tr>
            <td>relativer Luftdruck</td>
            <td><*var=press*> hPa</td>
      </tr>
      <tr>
            <td>Luftfeuchtigkeit</td>
            <td><*var=hum*> &#37;</td>
      </tr>
      <tr>
            <td>Windrichtung</td>
            <td><*var=winddir*> &#176;</td>
            <td><*var=dirstr*></td>
      </tr>
      <tr>
            <td>Windgeschwindigkeit</td>
            <td width="100"><*var=speed_m*> m/sec</td>
            <td width="90"><*var=speed_kmh*> km/h</td>
            <td width="90"><*var=speed_kn*> kn</td>
            <td width="90"><*var=speed_bf*> bft</td>
      </tr>
      <tr>
            <td>Taupunkt</td>
            <td><*var=dew*> &#176;C</td>
      </tr>
      <tr>
            <td>gef&uuml;hlte Temperatur</td>
            <td><*var=chill*> &#176;C</td>
      </tr>
      <tr>
            <td>Regen</td>
            <td width="80"><*var=rph*> l/h</td>
            <td width="100"><*var=rpd*> l/24h</td>
      </tr>
</table>
```

Template example

7 PHP files

There are several php scripts included as samples how to visualize the data from the log files.
The results can be seen on
http://www.ur9.de/graphs.shtml